

TEMPORARY CLOUD PROVIDER CREDENTIALS VIA SECURE DISCOVERY FRAMEWORK

CROSS REFERENCE TO RELATED APPLICATIONS

[001] This application claims the benefit of U.S. Provisional Application No. 63/087,155, filed on October 2, 2020, and U.S. Provisional Application No. 63/045,736, filed on June 29, 2020, both of which are hereby incorporated by reference herein in their entirety.

FIELD

[002] The field generally relates to supporting secure access to cloud provider resources in a multi-tenant software-as-a-service (SAAS) scenario.

BACKGROUND

[003] The study of genomic data can involve complex analysis by a variety of parties with differing expertise collaborating over time. Research typically starts with genomic data that may come from a variety of sources. The data can then be analyzed using a wide variety of techniques. Today's research projects can involve parties spread throughout the world who share data and/or collaborate on data analysis. While strides have been made in the field, and international standards for sharing genomic data have been developed, significant challenges to sharing genomic data still remain.

SUMMARY

[004] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

In one embodiment, a method comprises, in a computing system supporting a plurality of tenants accessing genomic computing services in a software-as-a-service platform that orchestrates access to genomic digital data resources via policy-based access control, discovering a cloud provider account for an identity accessing the software-as-a-service platform; sending a request to a credentials management service for limited temporary derived credentials valid for the cloud provider account; receiving the limited temporary derived credentials valid for the cloud provider account; and providing the limited temporary derived credentials for use by the identity.

In another embodiment, a multi-tenant, cloud-based system comprises one or more processors; memory coupled to the one or more processors; a mapping between identities accessing a software-as-a-service platform and cloud provider accounts; a policy store comprising policy-based access control definitions; a genomic digital data resource linked to a role identifier and stored at a given cloud provider account external to the software-as-a-service platform; wherein the memory comprises computer-executable instructions causing the one or more processors to perform operations comprising: based on the mapping, discovering a cloud provider account for an identity accessing the software-as-a-service platform; sending a request to a credentials management service for limited temporary derived credentials valid for the cloud provider account; receiving the limited temporary derived credentials valid for the cloud provider account; and providing the limited temporary derived credentials for use by the identity to access the genomic digital data resource.

In another embodiment, one or more computer-readable media comprise computer-executable instructions capable of causing a computing system to perform the following in a computing system supporting a plurality of tenants accessing genomic computing services in a software-as-a-service platform that orchestrates access to genomic digital data resources via policy-based access control: discovering a cloud provider account for an identity accessing the software-as-a-service platform; sending a request to a credentials management service for limited temporary derived credentials valid for the cloud provider account; receiving the limited temporary derived credentials valid for the cloud provider account; and providing the limited temporary derived credentials for use by the identity to access the genomic digital data resources at the cloud provider account according to the policy-based access control; wherein: the software-as-a-service platform supports multiple different cloud provider account types per tenant; and the software-as-a-service platform supports multiple different cloud provider accounts per tenant.

BRIEF DESCRIPTION OF THE DRAWINGS

[005] FIG. 1 is a block diagram of an example system implementing policy-based genomic digital data sharing.

[006] FIG. 2 is a flowchart of an example method of implementing policy-based genomic digital data sharing.

[007] FIG. 3 is a block diagram of an example system comprising a platform implementing policy-based genomic digital data sharing via a signed access token.

[008] FIG. 4 is a flowchart of an example method of implementing policy-based genomic digital data sharing via a signed access token.

- [009] FIG. 5 is a block diagram of an example system generating a signed access token based on an access request and a policy document.
- [010] FIG. 6 is a flowchart of an example method of generating a signed access token based on an access request and a policy document.
- [011] FIG. 7 is a visualization of supported scenarios.
- [012] FIG. 8 is a block diagram of an example policy document.
- [013] FIG. 9 is a block diagram of an example signed access token.
- [014] FIG. 10 is a block diagram of an example system generating an access token based on attributes of an access request and conditions of a policy document.
- [015] FIG. 11 is a flowchart of an example method of generating an access token based on attributes of an access request and conditions of a policy document.
- [016] FIG. 12 is a block diagram of a system publishing genomic content for policy-based sharing.
- [017] FIG. 13 is a flowchart of an example method of publishing genomic content for policy-based sharing.
- [018] FIG. 14 is a flowchart of an example method of accessing published, shared genomic content.
- [019] FIG. 15 is a flowchart of an example method of registering an external service provider.
- [020] FIG. 16 is a flowchart of an example method of integrating an external service provider into a policy-based sharing platform.
- [021] FIG. 17 is a block diagram of an example system validating a signed access token.
- [022] FIG. 18 is a block diagram showing integration of a smart contract into a policy-based sharing platform.
- [023] FIG. 19 is a flowchart of an example method of implementing a smart contract in a policy-based sharing platform.
- [024] FIG. 20 is a flow diagram of an example publishing use case.
- [025] FIG. 21 is a flow diagram of an example external service provider use case involving registration.
- [026] FIG. 22 is a flow diagram of an example external service provider use case involving integration.
- [027] FIG. 23 is a block diagram of an example computing system implementing temporary limited derived credentials.

[028] FIG. 24 is a flowchart of an example overall method of providing temporary limited derived credentials.

[029] FIG. 25 is a flowchart of an example method of managing cloud provider accounts to provide temporary limited derived credentials.

[030] FIG. 26 is a sequence diagram of an example implementation of a method of managing cloud provider accounts to provide temporary limited derived credentials.

[031] FIG. 27 is a flowchart of an example method of providing limited temporary derived credentials.

[032] FIG. 28 is a sequence diagram of an example implementation of a method of providing limited temporary derived credentials.

[033] FIG. 29 is a block diagram of an example cloud provider mapping.

[034] FIG. 30 is a screen shot of an example user interface providing limited temporary derived credentials.

[035] FIG. 31 is a block diagram of an example credential object for underlying credentials.

[036] FIG. 32 is a block diagram of an example identity hierarchy.

[037] FIG. 33 is a block diagram of an example computing system in which described embodiments can be implemented.

[038] FIG. 34 is a block diagram of an example cloud computing environment that can be used in conjunction with the technologies described herein.

DETAILED DESCRIPTION

Example 1 – Overview

[039] The ever-growing availability of genomic data presents new opportunities for research and analysis. Today's sequencing platforms can generate a wide variety of sequenced output, including whole-genome sequencing (WGS). Also, various organizations such as the Global Alliance for Genomics & Health have developed standards for sharing genomic data. However, in practice, today's genomic information ecosystem can appear fractured at times. Data may be segregated or segmented into silos due to a variety of considerations, including technical, security, legal, and financial reasons. And, even when data is publicly available, it may not be fully integrated in such a way that it is immediately useful.

[040] One large hurdle is sharing information among parties. A fully open platform that allows all participants to share every piece of data of every other participant is not realistic or desirable. However, a policy-based approach to sharing genomic digital data among software-as-a-service

tenants can allow parties to share data in a controlled way that encourages collaboration between parties. Public data can be included, and external service providers can also participate. Access control can be automated and more easily controlled without having to manually engage in lengthy, complex security administration.

[041] As a result, a cloud-based platform can serve as a virtual space in which parties from a wide variety of backgrounds and institutions can collaborate, sharing data, knowledge, tools, workflows, and applications to converge on innovative insights and arrive at new solutions.

[042] In addition, technologies described herein can support cloud provider accounts in a way that is easily maintained and transparent to users. As described herein, multiple cloud provider types can be supported in a single platform and for a single tenant. Cross-tenant scenarios can also be supported. Limited temporary derived credentials can be provided for accessing the appropriate cloud provider account.

[043] Released from technical limitations, data can migrate to where it is needed, and a more cooperative ecosystem can result. Because the technologies apply to genomic digital data generally, they can be applied across a large number of use cases involving genomic digital data storage, retrieval, and analysis.

Example 2 – Example System Implementing Policy-Based Genomic Digital Data Sharing

[044] FIG. 1 is a block diagram of an example system 100 implementing policy-based genomic digital data sharing. In the example, a plurality of tenants 110A-N with associated user identifiers 120 access an application hosting platform instance 135 running on a data center 130. The platform instance 135 comprises a platform authentication service 140, a plurality of hosted applications 150A-N, an administrative service 158, a policy store 160 (e.g., with policy documents as described herein), and authentication tokens 170. As described herein, some scenarios can involve trust documents (not shown) that can influence policy-based access to genomic digital data.

[045] The applications 150A-N as part of processing can access one or more genomic data services 190A-N, which typically provide genomic digital data.

[046] In practice, the systems shown herein, such as system 100, can vary in complexity, with additional functionality, more complex components, and the like. For example, plural data centers 130 can be implemented, and such data centers can implement plural application hosting platform instances 135. Additional components can be included to implement security, redundancy, load balancing, report design, and the like.

[047] The described computing systems can be networked via wired or wireless network connections, including the Internet. Alternatively, systems can be connected through an intranet connection (e.g., in a corporate environment, government environment, or the like).

[048] The system 100 and any of the other systems described herein can be implemented in conjunction with any of the hardware components described herein, such as the computing systems described below (e.g., processing units, memory, and the like). In any of the examples herein, the genomic digital data, policy documents, authentication tokens, and the like can be stored in one or more computer-readable storage media or computer-readable storage devices. The technologies described herein can be generic to the specifics of operating systems or hardware and can be applied in any variety of environments to take advantage of the described features.

Example 3 – Example Method Implementing Policy-Based Genomic Digital Data Sharing

[049] FIG. 2 is a flowchart of an example method 200 of implementing policy-based genomic digital data sharing and can be performed, for example, by the system of FIG. 1 (e.g., the application hosting platform instance 135).

[050] At 210, a new tenant 210 is on-boarded. As a result, the tenant is assigned a tenant identifier and given the ability to share genomic digital data via the tenant identifier. In practice, such on-boarding can be performed any time in advance of receiving requests to publish data for sharing and need not be considered part of the publishing/access scenario.

[051] At 220, the platform receives a request from the tenant to publish genomic digital data within the system, and the request comprises a policy document that controls sharing. In a computing system comprising a plurality of tenants seeking access to genomic digital data resources provided by one or more genomic data services in a software-as-a-service platform that orchestrates access to the genomic digital data resources via policy-based access control, a policy-based access control definition (e.g., the policy document) can be received for a first of the tenants for a given genomic digital data resource. The definition can be received from the first of the tenants or another party (e.g., in an external service provider scenario).

[052] At 240, a request to access the shared genomic digital data from another tenant is received. A request is received for access to the given genomic digital data resource from a second of the tenants seeking access to the given genomic digital data resource.

[053] At 250, the request to access the shared genomic digital data is granted based on (e.g., according to) the policy document as configured by the owning tenant (e.g., the tenant who shared the data). Access is granted based on the policy-based access control definition. As described

herein, tokens can be provided in request to the access request. The token can be generated based on the relevant policy and then used to control access to the data (e.g., using role identifiers as described herein). Access to the given genomic digital data resource can be controlled by a role identifier linked to the policy-based access control definition.

[054] In practice, a single party (e.g., operating the platform) may perform all the acts shown; however, it is also possible that one party only performs some actions (e.g., on-boarding) while another party performs others (e.g., granting). Division of tasks may also take place along domain lines (e.g., one party performs functions relating to publication, and another performs functions related to granting access).

[055] The illustrated actions can be interpreted from alternative perspectives while still implementing the technologies. For example, “receiving a request” can be also interpreted as “sending a request” from a tenant’s perspective.

[056] The method 200 and any of the other methods described herein can be performed by computer-executable instructions (e.g., causing a computing system to perform the method) stored in one or more computer-readable media (e.g., storage or other tangible media) or stored in one or more computer-readable storage devices. Such methods can be performed in software, firmware, hardware, or combinations thereof. Such methods can be performed at least in part by a computing system (e.g., one or more computing devices).

[057] When implemented in computer-readable media, the technologies can comprise computer-executable instructions capable of causing a computing system to perform respective of the method steps.

Example 4 – Example Genomic Digital Data

[058] In any of the examples herein, genomic digital data can be the subject of policy-based sharing. Such data can take the form of sequenced DNA, RNA, or the like (e.g., the output of a sequencer, which, in the case of DNA, typically takes the form of a digital representation of strands consisting of four types of nucleotides: adenine (A), cytosine (C), guanine (G), and thymine (T). The nucleotides can be represented digitally in a variety of ways and encodings, but typically have an equivalent string representation of A’s, C’s, G’s, and T’s that is used for convenience of description. Although DNA examples are given, RNA sequencing can be used as well. Similarly, the term “genomic” encompasses information from genomes, exomes, and transcriptomes.

[059] In practice, the sequence information is accompanied by other useful information for research, including substantive data, such as a source of the DNA (e.g., demographics of the

subject, pathologies of the subject, and the like). Disease and phenotype information can be included and/or associated with the genomic digital data. Sequencing metadata can be included as well (e.g., the machinery/instrument and technology used to sequence the DNA, the date the sequencing was done, sequencing yield, quality metrics, a pointer to the sequencing run records, and the like). Other metadata such as the name of the originating party, legal restrictions, and the like can also be included.

[060] To facilitate sharing, the data can be provided in a common format that allows for analytics and workflows to be used across tenants. Such formats can be proprietary or open formats to facilitate open exchange of information in sharing scenarios.

[061] Policy-based sharing can be extended to other genomic data, such as executable workflow definitions related to genomic digital data and the like. Thus, a tenant can access both the executable workflow for processing genomic digital data as well as the underlying data itself via the policy-based sharing technologies described herein. A shared executable workflow definition may come from one source (e.g., a tenant), while the underlying data comes from the same or a different source (e.g., a same or different tenant). Such executable workflows can relate to protocols that have been established for the sake of reliability, consistency of results, and the like. Thus, for a particular research project, a given executable workflow may be shared across participants. Custom executable workflows can be developed by tenants and shared as well.

[062] The executable workflow can be executed (e.g., interpreted) by an engine or service that interfaces with sequencing instruments, thereby greatly simplifying, automating, and increasing the reliability and repeatability of the sequencing process. Error recovery and other features can be incorporated into such executable workflows. Workflows can be aimed at a variety of sequencing and related analysis tasks, such as demultiplexing, mapping and aligning, position sorting, duplicate marking, variant calling, and the like. Specialized workflows devoted to tumor-only or tumor-normal modes can be designed for detecting somatic variants in tumor samples. Many other scenarios are possible.

[063] Due to the lengthy compute times and massive volumes of data, such workflows can be employed to deliver quickness, flexibility, and cost efficiency, enabling labs of various sizes and disciplines to take greater advantage of their genomic data. The sharing technologies described herein can greater leverage such data among tenants.

[064] For sake of convenience, shared genomic digital data is sometimes called a “resource” or “protected resource” to denote that the data is a resource to which access is controlled via a policy as described herein.

[065] Genomic digital data can be provided by a genomic data service. Such services can enforce access control and cooperate with the platform as described herein to acknowledge and validate access tokens.

Example 5 – Example Software Tenants

[066] In any of the examples herein, a variety of software tenants can be supported. A software tenant is sometimes called a “tenant” for sake of convenience. Such tenants typically take the form of an enterprise tenant, such as a corporation, governmental body, research institution or group, educational institution or group, organization of users, or the like. By taking advantage of the technologies described herein, such tenants can greatly benefit from policy-based sharing.

[067] Any given user of the platform can be assigned to a tenant. In a multi-tenant cloud system, users can share computing resources but have individualized, customizable user experiences and individual stored data. In practice, the tenant tends to represent a separate legal entity that has a separate agreement with the platform provider. Thus, a user identifier is typically associated with a single tenant, and services are provided to the user based on the agreement between the cloud provider and the tenant.

[068] Although the tenants may share computing resources administered by a cloud service provider, a distinguishing factor between tenants is that different tenants can have different subscriptions, different storage restrictions, and access levels to genomic digital data and services of the platform. Various other customizations can be done. Tenants are not necessarily application owners because the application owner can be the cloud provider or a third party. However, some tenants may develop their own applications.

[069] In a cloud-based scenario, a framework is provided transparently to users to leverage redundancy of functionality and process among the tenants. However, boundaries between the tenants can be enforced to prevent access by one tenant to another of the tenant’s data. Each tenant’s data can be isolated and remain invisible to other tenants. Such an arrangement is typically a basic characteristic of multi-tenant systems. However, while such isolation is typically desirable, for some data there are great benefits to allowing policy-based sharing between tenants as described herein.

[070] Therefore, while the described platform can bear the characteristics of a traditional cloud-based multi-tenant system, it can also allow controlled sharing between and among the tenants, including proxy tenants as described herein.

Example 6 – Example Proxy Tenants

[071] In any of the examples herein, a proxy tenant can be implemented. A proxy tenant can be registered as a tenant and have a tenant identifier, but the tenant identifier is not used in the capacity of usual tenant functionality, whether or not the represented entity is an actual tenant of the platform. For example, in a public sharing scenario, a proxy tenant can be set up for the public data, whether or not the source (e.g., owner) of data is actually involved (e.g., a website, government agency, foundation, or the like) because the data is public. The proxy tenant has a tenant identifier, and digital genomic digital data can be published under the tenant identifier. In this way, public sharing scenarios can be supported by the platform. In practice, a data-owning proxy tenant may have an actual tenant identifier as well. So, the organization may have multiple tenant identifiers, one in its capacity as a source of public genomic digital data, another in its capacity as a research institution that takes advantage of the usual tenant functionality, and the like.

[072] Similarly, in an external service provider scenario, a proxy tenant can be set up for the external service provider. The external service provider can be assigned a tenant identifier that can be used to access and upload data to the platform for sharing under the tenant identifier. In this way, external service providers can be supported by the system. Again, the external service provider may have an actual tenant identifier as well, which is used when the external service provider is acting in the capacity of a usual tenant.

[073] Finally, the platform administrator or other similar party can operate in the capacity of a tenant or tenant delegate to provide any of the tenant-based functionality described herein. Such an arrangement can be helpful when a tenant does not wish to become involved in the platform or is unavailable or unable to do so.

[074] So, in any of the examples herein, a tenant can be a proxy tenant, and the tenant identifier for such a tenant can be processed according to the technologies described herein to achieve policy-based sharing.

Example 7 – Example Roles

[075] In any of the examples herein, roles can be used to control access to shareable genomic digital data. As described herein, a role can be uniquely identified by a role identifier. Such a role identifier can be created when a tenant wishes to publish resources for sharing and be linked to the policy document for a given resource.

Example 8 – Example Role Binding

[076] To accomplish the technologies described herein, late binding of a role to a user can be implemented. In a late binding scenario, the user identifier (or tenant identifier of the user identifier) can be bound to a role identifier at execution time (e.g., when access to resources is requested, when a list of available resources is requested, etc.) instead of beforehand. In this way, role assignment can be dynamic in that if the policy changes, a role assignment can also change automatically. Thus, roles can change over time without explicitly specifying a particular user. The user's membership in a tenant or workgroup can cause role assignment to change if the policy references such attributes. As described herein, binding can take place at a time of the request and be based on the user identifier or tenant identifier of the request.

[077] Similarly, policy-based sharing means that any changes to the policy can result in changes to sharing (e.g., role assignment). Policies that further rely on other factors (e.g., agreement status, agreement level, subscription status, subscription level, or the like) can cause a change to role assignment if such factors change. For example, if a tenant acquires a new subscription level, users from the tenant can automatically be granted additional access because a role may be assigned at execution time the next time that a user of the tenant requests access.

[078] Thus, late binding of role and the dynamic nature of role assignment can support a wide variety of flexible, automatic scenarios that avoid individual role assignment to particular users in advance. Resources needed to administer the system are thus greatly reduced while providing such rich functionality.

[079] Additional roles can be provided at the application level (e.g., that are enforced by the application); such roles may have early or late binding. For example, an application role identifier may specify a lab manager, principal, assistant, or the like. The application role identifier can itself be used in a condition statement that controls access to the policy-based role. In such a case, responsive to determining that the access request has attributes indicating a role identifier meeting a condition specified in the policy document, the role is included with appropriate permissions (e.g., as specified for the role) in the signed access token.

Example 9 – Example Permissions

[080] In any of the examples herein, permission to share a resource can be specified by specifying a service type, resource type, and permission type (e.g., "GDS.FILES.UPDATE," "GSS.LIBRARYPOOLS.READ," or the like).

[081] Permission types can include, manage, archive, create, delete, destroy, download, hide, lock, read, update, write, admin, run, grant, or the like. Resource types can include subscriptions, files, sequencing runs, library pools, library prep kits, analysis versions, task versions, tasks, runs, workflows, and the like.

[082] Service types can include genomic data services, workflow execution services, and the like.

Example 10 – Example Platform

[083] In any of the examples herein, the infrastructure that provides policy-based sharing is sometimes called a “platform.” Such a platform can be integrated into a multi-tenant cloud-based platform that provides access to a plurality of applications by the tenants. As described herein, such a platform can become a virtual place where tenants can collaborate via the sharing functionality described herein.

[084] The platform can be implemented as a software-as-a-service (SaaS) platform that orchestrates access to the genomic digital data resources via the policy-based access control technologies described herein.

[085] Various portions of functionality can be referred to as being inside or outside of the platform, but either arrangement may be implemented. For example, some functionality can be delegated to other service providers or brought into the platform as desired. In some cases, functionality can be described as being in an authentication service platform, which can be separate from or integrated into the overall multi-tenant cloud-based platform.

Example 11 – Example System with Platform Implementing Policy-Based Sharing

[086] FIG. 3 is a block diagram of an example system 300 comprising a platform 350 implementing policy-based genomic digital data sharing via a signed access token 372.

[087] Although an application is not shown, in practice, the actual sharing functionality can be invoked by applications running on behalf of a tenant and requesting access to genomic digital data 397 via the platform 350 and supporting software.

[088] In the example, an owning tenant 310A controls access to the genomic digital data 397. Such control is accomplished by configuring the policy document 360 (e.g., by an administrative user interface). Configuration can include creating a custom role identifier 374 for inclusion in the policy document 360. Such configuration can be included as part of the publication process when the tenant 310A wishes to publish the data 397 for sharing. Although not shown, publication can also include generation of a signed grant token as described herein.

[089] Subsequently, when a tenant 310B wishes to access the data 397, it can do so by sending an request 320 to the platform 350. The policy document 360 controls generation of the signed access token 372 (e.g., which can also include the role identifier 374) as described herein.

[090] The tenant 310B can then send the access token 372 with the role identifier 374 to a genomic digital data service that provides access to the genomic digital data 397 based on the token 372.

Example 12 – Example Method of Implementing Policy-Based Sharing

[091] FIG. 4 is a flowchart of an example method 400 of implementing policy-based genomic digital data sharing via a signed access token and can be implemented, for example, by a system such as that shown in FIG. 3 (e.g., by the platform 350).

[092] In any of the examples herein, responsive to receiving a request for access, the role identifier specified in a policy-based access control definition (e.g., a policy document) can be provided for a request for access. The role identifier can then be included in a signed access token.

[093] In the example, at 410, configuration of a policy with a role identifier is received from the controlling (e.g., owner or delegate) tenant.

[094] At 420, a request is received from another tenant for access to data controlled by the policy configuration. As described herein, such request can involve a request for a token.

[095] At 430, the request is granted based on the policy as configured (e.g., a policy document as configured by the controlling tenant). For example, at 440 based on the policy, a signed access token (e.g., with role identifier) can be provided. At 450, the access request can be granted based on the signed access token (e.g., based on the presence of appropriate scope for the role identifier).

Example 13 – Example Owning Tenant

[096] In any of the examples herein, the term “owning tenant” can be used to express that the policy-based sharing is essentially tenant-to-tenant-based sharing. An owning tenant can grant access to genomic digital data for which it already has access rights. By publishing data and configuring a policy document, other tenants can then access the data of the owning tenant.

[097] In practice, the owning tenant can delegate sharing administration to another tenant, who can impersonate the tenant for sharing purposes. Thus, the owning tenant is sometimes called the “primary tenant.”

Example 14 – Example Access Request

[098] In any of the examples herein, an access request can take a variety of forms. For example, the access request can specify the genomic digital data that is desired to be shared (e.g., using an identifier). Alternatively, a general request can be sent, and a list of available resources and associated identifiers can be provided for selection. The access request can then be completed by providing the identifier of the specific resource desired.

[099] In practice, the access request can be provided via communication between an application and a platform providing policy-based sharing services.

[100] A signed access token can be received responsive to the request, and the token is used to actually control access to the protected resource.

[101] In a session-based system, an access request can be sent when the session begins (e.g., the user authenticates), and an access token can be generated based on the user identity. Applications spawned from the session then have access to the resources indicated by roles in the access token.

Example 15 – Example Invitation

[1012] In any of the examples herein, an invitation process can be used to invite tenants for sharing. For example, a newly on-boarded tenant may receive certain invitations by default. Other tenants may receive an invitation upon signing up for a particular application or service. For example, a subscription model for applications can provide access to an application (e.g., and any associated public data shared as described here) upon subscription to the application. Other tenants may receive an invitation as part of being added to a policy document.

[1013] In practice, invitations can be controlled by policy documents or other resources that indicate when sharing is initiated.

[1014] The invitation process can include legal compliance, identity verification, key exchange, trust delegation, and the like.

Example 16 – Example System Generating Signed Access Token

[1015] FIG. 5 is a block diagram of an example system 500 generating a signed access token 572 based on an access request 520 and a policy document 560.

[1016] In the example, a user identifier 505 of a given tenant is accessing an application 510 that requests access to underlying genomic digital data 597.

[0107] The access request 520 can include a plurality of attributes, and in the example comprises a user identity set 530, which includes a workgroup identifier 535, a tenant identifier 537, and an application identifier 540.

[0108] The platform authentication service token generator (e.g., of authentication service 140 of FIG. 1 or the platform 350 of FIG. 3), receives the access request 520 as input and generates the signed access token 572 based on the policy document 560. In the example, the user identity set 530 and application identifier 540 support access, so the token 572 includes the role identifier 574 and the tenant identifier 576 of the tenant at issue (e.g., of which the associated user is a user).

[0109] The genomic data service 590, when provided the signed access token 572, can validate the token and provide access based on presence of the role identifier 574, which can also be used in an access control list 595 of the genomic data service 590 providing access to the underlying data 597.

[0110] Further security can be provided via a grant token as described herein.

Example 17 – Example Method of Generating Signed Access Token

[0111] FIG. 6 is a flowchart of an example method 600 of generating a signed access token and can be implemented, for example, by a system such as that shown in FIG. 5 (e.g., by the token generator 550).

[0112] At 610, a request for access is received from a user identity set. In practice, an application being used by a user having the user identity set may actually send the request on behalf of the user identity of the user.

[0113] At 620, a role identifier is included in a signed access token based on the policy document associated with the protected resource and one or more attributes of the access request. For example, in a scenario where all tenants using a particular application are granted access, the role identifier can be included responsive to determining that the access request is coming from an instance of the application. If all tenants having the application are granted such access, user identity may not play a part in the decision. However, in a tenant-to-tenant sharing scenario, the tenant identity may be a controlling factor (e.g., the tenant identifier of the requesting tenant must match a condition specified in the policy document). The workgroup identifier of the request may or may not be a controlling factor depending on conditions specified in the policy document.

[0114] At 640, a signed access token with the role identifier is sent to a genomic data service.

[0115] Access can then be granted to the genomic digital data based on the role identifier.

Example 18 – Example Applications

[0116] As shown in FIG. 5, an application 510 may send access requests on behalf of the user identifier 505. In any of the examples herein, although a request may be described as coming from a tenant or user identifier, in practice, an application may be operating on behalf of such a tenant or user identifier. The application instance can be associated with an authenticated user identifier and/or tenant identifier, which is used for security purposes (e.g., authenticating requests, determining tenant identifier, determining workgroup identifier, and the like).

[0117] Such applications can take a variety of forms and can be used for the acquisition, management, and analysis of genomic digital data as described herein.

Example 19 – Example Supported Scenarios

[0118] FIG. 7 is a visualization 700 of supported scenarios 710 that can be implemented via the technologies described herein.

[0119] Public access sharing 720 can be implemented as described herein by publishing genomic digital data that is public or desired to be public under a tenant identifier that is used to configure a policy document stating that the data is available (e.g., to all users, all users of a given application, or some other criteria).

[0120] An example of public access sharing can be implemented with respect to an application. So, for example, any tenant who subscribes to a particular application can be granted access a collection of public data in a format compatible with the application. In such a case, the policy document can specify that requests from the application (e.g., “application:Olympia”) for all tenants (e.g., “tid:”) are granted access to the shared public data.

[0121] Tenant-to-tenant sharing 730 can be implemented as described herein by publishing genomic digital data under a tenant identifier that configures a policy document specifying conditions controlling sharing (e.g., which other tenants can access the data). Although an invitation process can be involved, the other tenants need not configure the role identifier because the controlling tenant can do so.

[0122] Workgroup-based sharing 740 can be implemented as described herein by publishing genomic digital data under a tenant identifier that configures a policy document specifying conditions controlling sharing (e.g., which one or more workgroups can access the data). Although an invitation process can be involved, the members of the workgroup need not configure the role identifier because the controlling tenant can do so. A workgroup can be intra- or inter-tenant (e.g., span across a plurality of tenants).

[0123] Sharing to/from an external service provider 750 can also be implemented as described herein by creating a special tenant identifier for the external service provider, even if they are not acting in the capacity as a tenant proper. In this way, external service providers can access genomic digital data on the platform, perform analysis on it and publish results back to the platform for access by the tenant (e.g., who requested that the external service provider perform the analysis).

[0124] Other scenarios are possible because the policy documents can include a rich set of conditions that permit sharing. Evaluation of the policy document at execution time can be used so that mass re-configuration of individual user roles by tenant administrators can be avoided.

Example 20 – Example Workgroups

[0125] In any of the examples herein, any number of users can be assigned to be members of a workgroup identified by a workgroup identifier within the platform. Such users can be of the same tenant or span across tenants. Membership in the workgroup can be controlled by an administrator or programmatic process.

Example 21 – Example Token Signatures

[0126] In any of the examples herein, an access or grant token can be digitally signed by the controlling tenant for authentication. In practice, a public-private key cryptography approach can be used, where the token is signed with the tenant's private key and authenticated with the tenant's public key.

[0127] In practice, the keys of a platform administrator or a delegate can be used in place of the tenant's key to simplify administration. Any keys that are trusted and verifiable by the platform can be used to achieve a trust relationship that is enforced to prevent unauthorized sharing between the tenants.

Example 22 – Example Policy Document

[0128] In any of the examples herein, a policy document can be used to control sharing. Such a policy document can thus serve as a policy-based access control definition. As described herein, the policy-based access control definition can be evaluated at a time a request for access is received.

[0129] FIG. 8 is a block diagram of an example policy document 860 that can be used in any of the examples herein. In practice, the policy document 860 is configured (e.g., created, read, updated, or deleted) by the tenant controlling the resource (e.g., genomic digital data) with which the policy

document 860 is associated in the platform. For example, an administrative user interface can be provided for access by an administrator user of the tenant or configuration can be done programmatically if desired.

[0130] As described herein, a policy document 860 can filter access requests based on application identifier or name, identity (e.g., tenant identifier, workgroup identifier, or the like), or the like. Although not shown, the policy document 860 can be linked (e.g., mapped) to a role identifier (e.g., controlled by the configuring tenant), and the policy document 860 thus accomplishes control over access by serving as a gatekeeper to the role identifier, which can ultimately be used to authorize access to the protected sharable resource.

[0131] A variety of formats can be used to accomplish filtering. In the example, the policy document 860 can include metadata 861 (e.g., a date, version, or the like) and one or more statements 862. The statements can take the form of an effect, a tenant identity parameter 863 and zero or more conditions 864. The effect can specify that the effect takes effect if the identity parameter 863 and the conditions 864, if any, are met. Such an effect can be that sharing is allowed (e.g., “allowed”) or that particular types of sharing are allowed (e.g., read-only, read-write, or the like), or permissions as described herein are granted; however, the type of sharing can alternatively be accomplished by creating different role identifiers with differing levels of access.

[0132] In practice, the identity 863 is listed separately to emphasize that a tenant identity parameter is typically specified as part of the policy document 860 and effectively serves as a condition. For example, a specific tenant, list of tenants, or wildcard can be listed as the tenant identity parameter. If a request comes in from a tenant identifier satisfying the tenant identity parameter, the parameter is considered satisfied, and the statements will be executed if any conditions 864 are also satisfied.

[0133] As described herein, a given condition 864 can include a filter parameter such as application identifier, workgroup identifier, application role identifier, or the like. A request that has attributes that satisfy the condition cause execution of the statement (e.g., which enables access). Thus, access to the underlying role can be filtered based on such attributes.

[0134] If the policy document is satisfied, then the linked role identifier is included in the generated access token as described herein.

[0135] Additional functionality or configuration can be incorporated into the policy document 860 as desired to extend sharing functionality. For example, a policy document 860 can incorporate or refer to trusted external resources such as smart contracts as described herein.

Example 23 – Example Signed Access Token

[0136] In any of the examples herein, a shared access token can be generated based on a policy document and the incoming access request to control access to the resource linked to the policy document (e.g., via a role identifier).

[0137] FIG. 9 is a block diagram of an example signed access token 972 that can be used in any of the examples herein. In practice, an actual token 972 can take different forms with more or fewer fields therein.

[0138] The subject 974 can be a system user identifier.

[0139] The issuer 976 can indicate which instance of the platform authentication system issued the token. Alternatively, the issuer can be the controlling tenant.

[0140] The tenant identifier 978 can indicate the tenant identifier of the user (e.g., the user associated with the application requesting the resource).

[0141] Membership can be encoded into the access token 972 based on the user roles and permissions as well as if they user satisfies the policy criteria. During access token generation, a user identifier that fulfils the policy criteria automatically gets the associated role identifier as the membership according to the policy set at the time of granting access. 980 can include the list of memberships that the user has access to. Membership can be indicated by a role identifier 982 and/or workgroup identifier 984. A permission index (or “*” for all) can be included. For example, a user can have membership in both a role and a workgroup.

[0142] The access control list 990 can include a tenant identifier and a user identifier along with granted permissions for the associated resources. The access control list in the token 972 can be included for efficiency purposes (e.g., so that a separate access control list need not be checked), or it can serve as a double check on an access control list already in place (e.g., an access control list already shipped to a genomic data service as part of the grant token).

[0143] The grant type 992 can indicate the grant type or authentication flow on how the user obtained the token 972.

[0144] The audience 994 can determine which cloud provider service the user is attempting to access.

[0145] The service 996 can indicate the application or service the user was using to generate the token 972.

[0146] The scope 998 can comprise a granted permissions list (e.g., identifiers indicating the granted type of access by specifying a service type, resource type, and permission type (e.g., “GDS.FILES.UPDATE,” “GSS.LIBRARYPOOLS.READ,” or the like)).

[0147] In practice, the signed access token 972 can be implemented as a JSON web token or other format supporting storage of the relevant fields. It can be signed with the signer's private key, allowing authentication via the signer's public key.

Example 24 – Example Access Token Generation System

[0148] FIG. 10 is a block diagram of an example system 1000 generating an access token 1072 based on attributes of an access request 1020 and conditions of a policy document 1060.

[0149] In the example, the access request 1020 can include a set of one or more attribute names 1040A-N – attribute value 1042A-N pairs. For example, attributes can indicate the tenant of the user identifier requesting access, a workgroup, an application associated with the request, or the like.

[0150] The policy document 1060 can include a role identifier 1074, which might not be explicitly stored in the document 1060, but instead be linked to it (e.g., in a mapping between role identifiers and policy documents). The policy document 1060 can include a plurality of conditions 1064A-N, including respective filter attribute 1064A – filter parameter 1066A pairs. The filter attribute 1064A can specify an attribute by name or identifier, and the filter parameter 1066A can specify a parameter that indicates which attribute values qualify for assignment of the role identifier 1074. In practice, the parameter 1066A can take the form of a single value, list, wildcard, or the like.

[0151] The access token generator 1050 can match the policy parameter to an attribute (e.g., of an incoming request). External conditions can also be included (e.g., conditions that are not part of the access request 1020).

[0152] If the access request 1020 qualifies for role assignment as indicated by the conditions 1064A-N, the role identifier 1074 can be included in the access token 1072, along with a tenant identifier (e.g., of the requesting user).

[0153] The token 1072 can be signed using a private key (e.g., of the controlling tenant or the cloud service provider). Such signing can be achieved using conventional or other public-private key cryptography methods and could be separate functionality from the token generator 1050. If signed, a public key of the signer can be used to authenticate the token 1072.

Example 25 – Example Access Token Generation Method

[0154] FIG. 11 is a flowchart of an example method 1100 of generating an access token based on attributes of an access request and conditions of a policy document and can be implemented, for

example by the system 1000 of FIG. 10 (e.g., the access token generator 1050 or other access token generating systems described herein).

[0155] At 1110, a request for access to shared genomic digital data is received, and the request comprises one or more attributes. Such attributes can take the form of attribute name – attribute value pairs, but the attribute names can be implied (e.g., based on position within the request or the like).

[0156] At 1120, a policy document for the shared genomic digital data is accessed, and the policy has one or more conditions.

[0157] At 1140, an access token is generated based on the one or more attributes of the request and the one or more conditions of the policy. For example, a role identifier can be included if the attributes indicate that the request meets the conditions of the policy. External attributes can also be included to influence generation of the token (e.g., whether the tenant of the requester has an increased subscription level or the like).

[0158] As described herein, the resulting token can be signed.

Example 26 – Example Genomic Digital Data Publishing System

[0159] FIG. 12 is a block diagram of a system 1200 publishing underlying data (e.g., genomic digital data) 1297 for policy-based sharing. In practice, the system 1200 can be incorporated into any of the policy-based sharing examples herein and be invoked to configure (e.g., set up) sharing.

[0160] In the example, a controlling tenant 1210 accesses a workgroup administrative console 1220 to provide access to shared underlying data 1297 provided by a genomic data service 1290.

[0161] An access control list 1292 can be created to enforce restrictions to the data 1297. The access control list 1292 can include an entry indicating the controlling tenant identifier 1234, the role identifier 1236 created for the given policy-based sharing scenario, and the granted permissions 1278 (e.g., indicating resource type, access type, or the like).

[0162] The tenant 1210 generates a policy document 1260 that is included in the policy store 1255 and linked with (e.g., mapped with) the tenant identifier of the tenant 1210, the role identifier 1236 and the underlying data 1297.

[0163] In the example, the policy document 1260 includes metadata about a version and a statement that allows access to all tenants (e.g., “TID:”) that are accessing the data via the application “Olympia.” A signed grant token 1230 is created that includes the one or more access control lists dictated by the publishing scenario. In this way, the access control lists can be shipped to the genomic data service 1290, where they are stored for future reference (e.g., to grant

permissions based on requests associated with the role id 1236). In the example, the tenant identifier 1234 of the controlling tenant and the role identifier 1236 created for the policy-based sharing scenario are included.

[0164] The illustrated scenario is sometimes called “publishing” the data (e.g., data 1297) because the tenant 1210 has made the data available to those who qualify (e.g., by those requests meeting the conditions in the policy 1260).

Example 27 – Example Genomic Digital Data Publishing Method

[0165] FIG. 13 is a flowchart of an example method 1300 of publishing genomic content for policy-based sharing and can be implemented, for example, by the system 1200 of FIG. 12 (e.g., the workgroup admin counsel 1220 or other part of the platforms supporting policy-based sharing described herein). As described herein, the method 1300 can be driven by the tenant granting access (e.g., a workgroup administrator).

[0166] At 1320, a custom role identifier is created (e.g., along with a policy document linked to the role identifier). Such a role identifier can be unique within the platform and is allocated in response to the publication request.

[0167] At 1340, a signed grant token is created with a list of access control lists. The grant token can be associated with (e.g., linked to) a resource identifier that identifies the genomic content as described herein.

[0168] At 1360, the content is published to the genomic data service with the grant token. For example, the data can be uploaded to the genomic data service if not already present. The grant token can be validated to control access to the protected resource.

Example 28 – Example Genomic Digital Data Accessing Method

[0169] FIG. 14 is a flowchart of an example method 1400 of accessing published, shared genomic content and can be implemented, for example, by any of the systems supporting policy-based sharing described herein. Such a method is typically driven by the enterprise user identifier accessing the resource.

[0170] At 1420, a request for access is received (e.g., by the platform from an accessing user identifier of a given accessing tenant).

[0171] At 1440, a signed access token is generated as described herein (e.g., based on the policy).

[0172] At 1460, the genomic digital data resource is accessed with the signed access token. For example, requests can be sent to a genomic data service, and the genomic data service responds with the data.

Example 29 – Example External Service Provider Registration Method

[0173] FIG. 15 is a flowchart of an example method 1500 of registering an external service provider and can be implemented, for example, by any of the systems supporting policy-based sharing described herein. Such a method 1500 is typically driven by an administrative user identifier or process. As described herein, a variety of external service provider scenarios can be supported.

[0174] At 1520, a registration of the external service provider with the platform is received (e.g., by the platform). Such registration can include scopes and grants of access and can be performed by an administrative user.

[0175] At 1540, registration of the external service provider as a proxy tenant is received. A tenant identifier can be used for the proxy tenant, even though the external service provider may not be acting in the capacity of a tenant or participate as a full tenant of the platform.

[0176] At 1560, policy-based access control is created (e.g., tenant-to-tenant sharing is enabled via a role created under the proxy tenant of the external service provider). A policy can be associated with the role. In practice, the data is considered to be owned by the external service provider (via the proxy tenant identifier), and the data is shared with accessing tenants via policy-based sharing as described herein.

[0177] A more detailed use case is described in FIG. 21 below.

Example 30 – Example External Service Provider Integration Method

[0178] FIG. 16 is a flowchart of an example method 1600 of integrating an external service provider into a policy-based sharing platform and can be implemented, for example, by any of the systems supporting policy-based sharing described herein. Such a method 1600 is typically driven by an accessing user identifier (e.g., from another tenant) or process.

[0179] At 1620, a workflow is launched that communicates with the external service provider. Such a workflow can be kicked off to perform tasks associated with the external service provider. For example, a tenant may have sent out a physical biosample and wishes to receive digital genomic data results of analysis of the biosample, a tenant may have generated genomic digital data

such as sequencing results and wish to have the results interpreted by the external service provider, or the like.

[0180] At 1640, a grant token is generated for the external service provider (e.g., for the particular sharing scenario). In practice, a workflow execution service executing the workflow can request generation of the grant token.

[0181] At 1660, the external service provider is called with the grant token, which is validated (e.g., using the administrative public key).

[0182] At 1660, results (e.g., of biosample analysis, data analysis, or the like) are received from the external service provider and accepted into the genomic data service (e.g., where they can be accessed by users of the tenant who initiated the workflow involving the external service provider). For example, the external service provider uploads the results to the genomic data service using a signed access token that was provided by or on behalf of the requesting tenant.

[0183] A more detailed use case and sample policies are described in FIG. 22 below.

Example 31 – Example External Service Provider

[0184] In any of the examples herein, an external service provider can be a service provider that provides a genomic data service to tenants of the system. Thus, a tenant for which a policy-based access control definition is received can be a proxy tenant representing an external service provider for which policy-based sharing is implemented. Because the external service provider is operating external to the system (e.g., not as a tenant of the system), a proxy tenant identifier can be set up for use by the external service provider, and the external service provider can be registered with the platform as associated with the proxy tenant identifier. As described herein, the external service provider can then take advantage of the policy-based tenant-tenant sharing technologies described herein.

[0185] Such service providers can perform useful services such as analyzing a physical biosample and uploading analysis results (e.g., digital genomic data), analyzing genomic data (e.g., using mathematical processes, machine learning, and the like), and the like.

[0186] From a user perspective, an external service provider can appear as a third-party application, the services of which are available to users. In this way, a rich research ecosystem can be provided where third-party applications can be interfaced into the platform so that the platform is not limited to only those applications provided by the platform orchestrator or other tenants.

Example 32 – Example Token Validation

[0187] FIG. 17 is a block diagram of an example system 1700 validating (e.g., authenticating) a signed access token that can be implemented to achieve token authentication in any of the examples herein. In the example, the signed access token 1772 (with role identifier 1774 and tenant identifier 1776) is signed with the private key of the controlling tenant 1710. In practice, the private key of the controlling tenant can be administered by the tenant or an administrator of the cloud service provider (e.g., the platform orchestrator).

[0188] The authenticator 1780 can accept the public key of the controlling tenant 1710 and the signed token 1772, and output an authentication result 1790 (e.g., whether or not the token 1772 was indeed signed by the private key of the controlling tenant 1710). The authenticator 1780 can take the form of conventional public-private key cryptography algorithms (e.g., including hashing and the like) to accomplish validation of the token 1772.

[0189] After validation, further processing can be performed to determine whether permissions are available for a given resource (e.g., based on memberships such as role identifier, workgroup identifier, and the like). Responsive to determining that the memberships satisfy specified conditions (e.g., satisfy the access control list), the associated permissions (e.g., in the access control list) are granted to the requestor associated with the token.

[0190] Although a signed access token 1772 is shown, the system 1700 can also be used for signed grant tokens described herein.

Example 33 – Example Genomic Data Implementations

[0191] In any of the examples herein, genomic data can take the form of a genomic file types. Such file types can be associated with different genomic data, differentiating between that acquired during sequencing of a genome (e.g., raw data from a sequencing instrument, the assembled genome, and the like), data for assistance during assembly (e.g., a reference genome), as well as data indicating results of comparative genomic analysis. Comparative genomic analysis can include comparison between or among genomes (e.g., files types that indicate single nucleotide polymorphisms, insertions, deletions, structure variants, and copy number variation within a genome as compared to a reference genome).

[0192] An example of such a file type is the VCF (SNP) file type. VCF stands for “Variant Call Format.” It is a standardized text file format for representing SNP, INDEL, SV, and CNV variation calls. SNPs (Single Nucleotide Polymorphisms) are the most common type of genetic variation

among the genomes of people. Each SNP represents a difference in a single DNA building block (e.g., nucleotide). In practice, this is a widely used VCF.

[0193] Another example of a file type is the VCF (INDEL) file type. Indel is a molecular biology term for insertions or deletions in DNA. The number of INDELs in human genomes is second only to the number of SNPs. INDELs can play a key role in genetics.

[0194] Another example is the VCF (SV) file type. SVs (or Structural Variants) are large DNA sequences that are inserted, inverted, deleted or duplicated within genomes.

[0195] Another example is the VCF (CNV) file type. A CNV (or Copy Number Variation) is when the number of copies of a particular gene varies from one individual to the next. Some cancers are believed to be associated with elevated copy number of particular genes.

[0196] Another example is the BAM file type. The Binary Alignment Map (BAM) can be the comprehensive raw data of genomic sequencing; it can include the lossless, compressed binary representation of the sequence alignment map. BAM files tend to be about 90-100 gigabytes in size. They can be generated by aligning the FASQ files to the reference genome. A BAM file (.bam) is the binary version of a SAM file. A SAM file (.sam) is a tab-delimited text file that contains sequence alignment data.

[0197] Another example is the FASTQ file type. FASTQ files contain billions of entries and are about 90-100 gigabytes in size, making them too large to open in a normal text editor. FASTQ files can be the ultimate raw data.

[0198] Another example is a quality control metric file type (e.g., report). Before running any alignment or assembly, it is possible to check the quality of the underlying data. Quality can be checked from within a sequencing program. A quality control analysis can test a number of different metrics and produce a consolidated report. The report can include a simple categorization (e.g., red, yellow, green) to indicate whether results are bad, intermediate, or good.

Example 34 – Example Specialized Permissions

[0199] In any of the examples herein, specialized permissions for a genomic context can be implemented. For example, permission granularity can be extended to the file type in policy statements. Thus, a policy can specify that different tenants, workgroups, users, or application roles can have different permissions for different genomic file types or different genomic file type categories (e.g., raw sequencing data, assembled genome, reference genome, comparative genomic analysis, or the like).

[0200] A specialized so-called “background” permission can allow utilization of the resource (e.g., file type) by applications or other infrastructure without granting read access (e.g., so it is not able to be read directly). For example, granting background permission to a reference genome, allows the reference genome to be used to assemble raw data, determine single nucleotide polymorphisms, or perform other comparative genomics analysis without granting read access to the reference genome itself.

[0201] In addition, specialized permissions can be specified for executable workflows. For example, a “high level run only” permission can allow high level visibility of the workflow (e.g., steps, progress of steps, error messages, and the like), without revealing the details of the workflow (e.g., the underlying interpreted code) or allowing modification of the workflow. Thus, a workflow can be shared among tenants without revealing all minor technical details within it.

Example 35 – Example Application Implementations

[0202] In any of the examples herein, an application can be devoted to facilitating genomics use cases, such as clinical genomics. For example, a cloud-based in vitro diagnostics solution for oncology can be built into an application that supports sample accessioning, wet lab protocol (e.g., extraction, library preparation, indexing/pooling), sequencing, demultiplexing, sequencing quality control, and then secondary analysis, eventually resulting in a report. Secondary analysis can include comparative genomic analysis, such as detecting single nucleotide variants and the like.

[0203] Such an application can coordinate a variety of services and unify management of genomic data to allow efficient and accurate collection and analysis of genomic data. For example, a genomic lab service, a workflow service, an event notification service, a task service, and a genomic data store can work in tandem under orchestration of an application that operates in the sharing environment described herein.

[0204] Accordingly, different actors working as tenants or external service providers can collaborate and share information using the described policy-based genomic data sharing technologies described herein.

Example 36 – Example Smart Contract Integration

[0205] FIG. 18 is a block diagram showing integration of a smart contract 1865 into a policy-based sharing platform that can be implemented to extend policy document functionality in any of the examples herein. A policy can refer to a contract so that whoever satisfies the contract gets access to the data via the policy; conversely, breach or absence of contract satisfaction means that the

party does not get access to the data via policy. The party can be specified as a tenant, workgroup, or the like.

[0206] In the example, the platform authentication service token generator 1850 consults a policy 1860 to determine how to generate the signed access token 1872 with role identifier 1874 and tenant identifier 1876. As described herein, the generator 1850 can also consult one or more attributes of the incoming request (e.g., a tenant identifier, application identifier, or the like).

[0207] As shown, the policy document 1860 can include or reference a smart contract 1865. The smart contract 1865 can itself make reference to a blockchain service 1897 that memorializes agreements for one or more tenants 1810A-N. Such agreements can be between the tenants, between the tenant and the cloud service provider, between the tenant and a third party, or some combination thereof. Such a blockchain service can make use of blockchain techniques such as consensus-based immutable recordation of agreements (e.g., agreement presence, agreement level, service level, or the like) and be built on blockchain infrastructure from any of a variety of providers or technologies (e.g., Ethereum-based functionality or the like).

[0208] Trust relationships between the platform and the service, tenant to tenant, and the like can be established via trust documents that can facilitate automated evolution of the policy document 1860 based on agreements indicated by the service 1897.

[0209] In this way, whichever tenant satisfies the terms of the contract gets access to the data specified in the associated policy. Automated contract administration is thus provided, facilitating immediate access to the data as specified by the contract upon satisfaction of the contract terms (e.g., payment, subscription, or other terms).

[0210] A grant token can be generated based on the completion of a contract, and the access token can be generated when access is requested, in light of the associated policy.

[0211] As a further feature, accesses to data can be logged for subsequent auditing functionality. Such logs can indicate the date and time of access, identifier of requesting party, identifier of the granting party, and the policy that permitted access, which itself can be annotated with compliance or legal reasons (e.g., "Agreement of 12/15/2017 between Party X and Party Y") or the like.

Example 37 – Example Trust Documents

[0212] In any of the examples herein, a policy-based sharing platform can document trust relationships between tenants as trust documents. For example, a trust document can store a consent agreement for one tenant reflecting that trust has been established with another tenant (e.g., by storing an origin tenant, destination tenant, consent agreement date, and consent metadata).

[0213] Such a trust document can be enforced as a prerequisite to sharing data with the tenant. For example, in such a scenario, a policy only takes effect if supported by a trust document.

Example 38 – Example Smart Contract Method

[0214] FIG. 19 is flowchart of an example method 1900 of implementing a smart contract in a policy-based sharing platform that can be implemented to extend policy document functionality in any of the examples herein.

[0215] At 1920, tenant agreements are reflected in a blockchain service (e.g., provided according to an Ethereum or other blockchain infrastructure).

[0216] At 1940, a request to access data controlled by one or more of the agreements is received. For example, a policy with reference to the agreement can be in place for the data.

[0217] At 1960, the request to access the data is granted based on the policy with reference to the blockchain service.

[0218] At a subsequent point in time, the blockchain service can be updated to reflect an agreement change for the tenant. As a result, the request may no longer be granted, may be newly granted, or the like. In other words, changes to the agreement can result in changes to whether access is granted based on the policy that references the agreement.

Example 39 – Example Publishing Use Case

[0219] FIG. 20 is a flow diagram of an example publishing use case 2000 that can be implemented in any of the examples herein. Although the example shows “public_access,” such a use can cover both public and tenant-to-tenant sharing and can parallel the description of the methods of FIGs. 13 and 14.

[0220] The initial phase of publishing resources with access control lists can be driven by an administrative user identifier or process. The controlling tenant 2010 can interact with the identity and access management console 2050, the platform 2052, and the genomic data service 2054 to accomplish publication of the public content 2060.

[0221] The subsequent phase of retrieving the resources can be driven by a user identifier from another tenant 2020. The access token can include membership (e.g., a role identifier can indicate membership). Receiving access can take the form of receiving a list of resources from which a selection can be made for actual access.

Example 40 – Example Grant Token

[0222] In any of the examples herein, a grant token can associate a role identifier with a resource (e.g., a resource identifier). The role identifier serves as a policy identity that contains the policy or rules for data access to the associated resource.

[0223] For example, a resource identifier can be included in the grant token, associated in a table that maps the grant token to the resource identifier, or otherwise linked to the grant token.

Example 41 – Example External Service Provider Use Case

[0224] FIGs. 21 and 22 are flow diagrams of an example external service provider use case method 2100, 2200 that can be implemented in any of the examples herein. Such a use case can parallel the description of the methods of FIGs. 15 and 16. First, the external service provider is registered (e.g., as a proxy tenant), and then the external service provider is integrated into the system (e.g., policy-based sharing is used to allow an external service provider access to the system, whether read access, write access, or both).

[0225] In an external service provider scenario, a single policy can accomplish sharing as described herein. Such a policy can be defined during registration of the external service provider into the platform. Such a policy can comprise information such as with which tenants the external service provider can share data. For example, in a scenario where the external service provider uploads data, the policy can both allow the external service provider to upload the data and allow the accessing tenant to access the data uploaded by the external service provider.

[0226] Data generated by the external service provider can go to a dedicated tenant (e.g., “tenant_ESP”), and the platform administrator can define a policy for the dedicated tenant to share data with a tenant that wishes to use external service provider sharing. When the accessing tenant generates the access token, the token is encoded with memberships based on their access rights and the role identifier specified in the policy is dynamically populated in as one of the memberships if the tenant fulfils the policy criteria.

[0227] The sharing scenarios can be used to support workflows involving the external service provider. Typical workflows that can be initiated are for the external service provider to upload genomic results from analysis (e.g., of a physical biosample), the external service provider to download genomic data and upload results of analyzing the genomic data (e.g., download genomic data, process the genomic data externally, and upload the analysis results), and the like. For example, a tenant may wish to take advantage of an external service provider that generates a variant report based on output from a sequencing process (e.g., sample files that contain base call

and quality information for reads passing filtering, such as FASTQ files). A tenant can run a workflow with the external service provider to upload the sample files to the external service provider. After upload, the external service provider can run their process and generate a variant file to which the tenant then has access.

[0228] The platform need not be aware of the inner workings of the external service provider. An input file can be sent, and the external service provider generates an output file, which is shared with the originating policy (e.g., rid:<>) when the file was initially uploaded. In the example, the external service provider can both read and write to the resource (e.g., file storage area).

[0229] The initial phase of registering the external service provider with the platform is shown in FIG. 21 and can be driven by a workgroup administrator identifier or process. The administrator user identifier 2110 can interact with the platform 2152 and the external service provider 2156. The workflow execution service 2153 and the genomic data service 2154 can enter in at a later time (e.g., integration, access, or both). Although the administrator user identifier 2110 may be for that of an administrator of the platform, a tenant administrator identifier may be awarded such authority (e.g., to register and integrate external service providers) if desired.

[0230] Subsequently, after registration, integration of the external service provider 2156 can be provided as part of the workflow that involves the services of the external service provider 2156. In practice, the data can be owned by a proxy tenant for the external service provider 2156 and shared with other tenants.

[0231] In the example, a platform administrator user identifier 2110 registers the external service provider 2156 (e.g., scopes and grants for the external service provider 2156) with the platform 2152.

[0232] The administrator user identifier 2110 then registers the external service provider 2156 with a dedicated tenant (e.g., a proxy tenant such as “Tenant_ESP” for the external service provider 2156). In a data writing scenario, external-service-provider-processed data can be streamed to the dedicated tenant, even though the external service provider may not be a full tenant of the system.

[0233] The platform administrator user identifier 2110 can then create policy-based access control that enables tenant-tenant data sharing. For example, the proxy tenant can share data with one or more specified tenants.

[0234] An example policy that allows the external service provider (“Tenant_ESP”) to share its data with Tenant1 is as follows:

[0235] rid:<tenantESP_tenant1_GUID> (Data Owned by: Tenant_ESP but data shared with tenant1 with limited permission Ex:GDS.FILES.READ)

```
[0236] {
[0237]   "Version": "1558387292",
[0238]   "Statement" : [
[0239]     "Effect" : "allow",
[0240]     "scope" : "GDS.FILES.READ"
[0241]     "Identity": { tid: tenant1 }
[0242]   ]
[0243] }
```

[0244] The policy is associated with the role identifier “tenantESP_tenant1_GUID.”

[0245] An example policy that allows the external service provider (tenant_ESP) to share its data with the Tenant1_Clinical_Workgroup is as follows:

[0246] rid:<tenantESP_tenant1_GUID> (Data Owned by : Tenant_ESP but data shared with tenant1 with limited permission Ex:GDS.FILES.READ)

```
[0247] {
[0248]   "Version": "1558387292",
[0249]   "Statement" : [
[0250]     "Effect" : "allow",
[0251]     "scope" : "GDS.FILES.READ"
[0252]     "Identity": { wid: tenant1_clinical_workgroup1 }
[0253]   ]
[0254] }
```

[0255] The policy is associated with the role identifier “tenantESP_tenant1_GUID.”

[0256] After completion of registration, integration can be implemented as shown in FIG. 22, which involves the same parties and a user identifier 2220 from an accessing tenant (e.g., Tenant1) that wishes to take advantage of the services offered by the external service provider 2256. In the example method 2200, the user identifier 2220 from the accessing tenant launches a workflow execution task (e.g., that communicates to the external service provider 2256) with the workflow execution service 2253. For example, the task might be entitled “Perform interpretation.” In the example, the external service provider 2256 provides results to the accessing tenant, where providing the results comprises uploading the results to the genomic data service 2254, where the accessing tenant can access them.

[0257] The workflow execution service 2253 sends a request to generate a grant token for the external service provider 2256 using the proxy tenant identifier (e.g., “Tenant_ESP”). The token

includes the access control lists per the policy. The platform 2252 responds with a grant token, which can take the following general form:

[0258] issuer = platform

[0259] audience = esp

[0260] access control list = [rid:<>]

[0261] tenant id = tenant1

[0262] membership = { }

[0263] The workflow can then call the external service provider 2256 with the grant token, which can be validated by the external service provider 2256 (the token's intended audience) using a public key of the platform orchestrator or other entity authorized to perform registration.

[0264] The external service provider 2256 can then send a request to the platform 2252 to generate an access token for the genomic data service 2254, copying the access control lists from the grant token's access control list claim. The platform 2252 responds with an access token, which can take the following general form:

[0265] issuer = platform

[0266] audience = gds

[0267] access control list = [rid:<>]

[0268] membership = { "rid":<> }

[0269] tenant id = tenant_ESP

[0270] The external service provider 2256 can then upload the results to the genomic data service 2254 using the access token, which can be validated by the genomic data service 2254 (the token's intended audience).

[0271] Subsequently, the uploaded data can be accessed by the user id 2220 of the accessing tenant (tenant1) or any other user of the accessing tenant if the user id has the appropriate membership (e.g., rid:<tenantESP_tenant1_GUID>) enabled via policy by the other tenant's administrator user.

[0272] An example policy that allows all users in the accessing tenant (tenant1) to see the processed data from the external service provider is as follows:

[0273] rid:<tenant1_ESP_data_read_access_GUID>

[0274] {

[0275] "Version": "1558387292",

[0276] "Statement" : [

[0277] "Effect" : "allow",

[0278] "scope" : "GDS.FILES.READ"

[0279] "Identity": { uid:* }

[0280]]

[0281] }

[0282] The policy is associated with the role identifier tenant1_ESP_data_read_access_GUID.

[0283] Access to the data uploaded by the external service provider is thus achieved by using the tenant-tenant policy-based sharing technologies described herein, where the external service provider is assigned a proxy tenant identifier.

[0284] The creator of a policy that has permission rights to resources can thus enable access to any internal or external tenants for a list of identities and the resources.

Example 42 – Example Version Field of Policy

[0285] In any of the examples herein, the version field of a policy can be used to facilitate audit tracking and rollback policies to a previous version.

Example 43 – Example Policies

[0286] In any of the examples herein, policies can be used to control sharing. Different policies can be used to achieve different sharing objectives. In the following example, a platform orchestrator "Illumina" maintains a platform that supports a variety of policy-based sharing scenarios.

[0287] Policies can be associated with role identifiers that ultimately control access to the shared resources. A policy can contain one or more identities (e.g., user identifier, application identifier, workgroup identifier, group identifier), scopes (e.g., permissions), and a role identifier (e.g., one policy can nest another policy). The policy (rid) can be associated with resources or identities to allow access to the resources.

[0288] For example, the following policy can achieve application-enabled content that allows any user that is using a particular application ("Olympia") to access content:

[0289] rid:<illumina_app_enabled_data> (Owned by : Illumina)

[0290] {

[0291] "Version": "1558387292",

[0292] "Statement" : [

[0293] "Effect" : "allow",

[0294] "scope" : "GDS.FILES.READ, GDS.FOLDERS.READ, GDS.FOLDERS.WRITE "

[0295] "Identity": { tid: *} }

```

[0296]     "Condition" : {
[0297]         "filter" : { "appid": "olympia"},
[0298]         "filetypes": ["sam","vcf","bam"]
[0299]     }
[0300] ]
[0301] }

```

[0302] The policy achieves application-enabled content by including a filter that specifies an application identifier of the application in question. Another filter limits access to certain filetypes specified in a filetype filter (e.g., sam, vcf, bam). As shown, the policy is associated with the role identifier “illumina_app_enabled_data”.

[0303] In another example, a policy allows public content to be shared with anonymous users who are using a specified application:

```

[0304] rid:<illumina_public_data> (Owned by : Illumina)
[0305] {
[0306]   "Version": "1558387292",
[0307]   "Statement" : [
[0308]     "Effect" : "allow",
[0309]     "scope" : "GDS.FILES.READ"
[0310]     "Identity": { tid: *}
[0311]     "Condition" : {
[0312]       "filter" : { "appid": "olympia"}
[0313]     }
[0314]   ]
[0315] }

```

[0316] The policy achieves read-only sharing with any user by specifying a read-only scope and including a wildcard for the tenant identifier. In the example, access is limited to those users using the application (“olympia”) specified in the application identifier filter of the policy. However, removing the application filter in the policy would allow read-only access by any user. As shown, the policy is associated with the role identifier “illumina_public_data”.

[0317] In another example, private content is shared with labs (workgroup) lab001 and lab002:

```

[0318] rid:<illumina_private_shared_data> (Owned by : Illumina)
[0319] {
[0320]   "Version": "1558387292",

```

```

[0321] "Statement" : [
[0322]     "Effect" : "allow",
[0323]     "scope" : "GDS.FILES.READ"
[0324]     "Identity": { wid: lab001, wid: lab002 }
[0325] ]
[0326] }

```

[0327] The policy achieves read-only sharing with any user in the two workgroups by specifying a read-only scope and including an explicit list of one or more workgroups. As shown, the policy is associated with the role identifier “illumina_private_shared_data”.

[0328] In another example, a tenant user 1 shares data with tenant 2 users having the user identifier “2”:

```

[0329] rid:<tenant1_private_shared_data> (Owned by : Tenant1's user- uid:1 )
[0330] {
[0331]     "Version": "1558387292",
[0332]     "Statement" : [
[0333]         "Effect" : "allow",
[0334]         "scope" : "GDS.FILES.READ"
[0335]         "Identity": { uid: 2 }
[0336]     ]
[0337] }

```

[0338] In the example, the policy achieves read-only sharing with a particular user identifier by specifying a read-only scope and by specifying the user identifier in the identity field. As shown, the policy is associated with the role identifier “tenant1_private_shared_data”.

[0339] In another example, a workgroup in tenant1 shares data with a user in the tenant2 with restricted permissions (i.e., read files and write files only):

```

[0340] rid:<tenant1_workgroup1_private_shared_data> (Owned by : Tenant1's Workgroup owner)
[0341] {
[0342]     "Version": "1558387292",
[0343]     "Statement" : [
[0344]         "Effect" : "allow",
[0345]         "scope" : "GDS.FILES.READ, GDS.FILES.WRITE"
[0346]         "Identity": { uid: 2 }
[0347]     ]

```

[0348] }

[0349] In the example, the policy is associated with the role identifier

“tenant1_workgroup1_private_shared_data”.

[0350] In yet another example, a workgroup in tenant1 shares data with another workgroup in tenant2:

[0351] rid:<tenant1_workgroup1_private_shared_data> (Owned by : Tenant1's Workgroup owner)

[0352] {

[0353] "Version": "1558387292",

[0354] "Statement" : [

[0355] "Effect" : "allow",

[0356] "scope" : "GDS.FILES.READ, GDS.FILES.WRITE"

[0357] "Identity": { uid: 2, wid: lab002 }

[0358]]

[0359] }

[0360] In the example, the policy is associated with the role identifier

“tenant1_workgroup1_private_shared_data”, which is reused from the previous example. Thus, more than one policy can be associated with a role identifier, allowing stacked policies that can be used to expand access in practice (e.g., a policy can be re-used across role identifiers to grant similar users access to different resources).

[0361] As shown, a variety of policies can support a variety of sharing scenarios.

Example 44 – Example Security Context

[0362] In any of the examples herein, a role identifier (e.g., role ID, rid, or the like) can alternatively be implemented as a security context identifier (e.g., context ID, cid, or the like).

Example 45 – Example Collaborating Parties

[0363] In any of the examples herein, parties can collaborate on the platform by sharing genomic digital data. As described herein, such parties can be workgroups, tenants, or both. Collaborating workgroups can be intra-tenant workgroups (e.g., one tenant) or inter-tenant workgroups (e.g., one or more workgroups of a tenant collaborate with one or more workgroups of another, different tenant). Parties can include patients, research laboratories, clinical laboratories (e.g., Quest Diagnostics, LabCorp, or the like), contract laboratories, medical clinics, hospitals, universities, specialists, counselors (e.g., genetic counselors or the like), companies, genomic services

companies (e.g., 23AndMe, Ancestry, or the like), agencies, (e.g., U.S. Centers for Disease Control and Prevention, U.S. Food and Drug Administration, European Medicines Agency, China Food and Drug Administration, World Health Organization, and the like), and the like.

Example 46 – Example Use Cases

[0364] The technologies described herein can be used in any of a wide variety of scenarios implemented on genomic information processing environments and platforms. For example, the technologies can support primary analysis, secondary analysis, and tertiary analysis workflows within or across collaborating parties. In addition to intra-analysis collaboration, cross-analysis collaboration can also be supported whereby a feedback loop of tertiary analysis results can be provided back to a party that performed secondary analysis for re-computation of the secondary analysis based on the tertiary analysis results. The technologies described herein can also be used to enforce research use only restrictions or to limit use for diagnostic purposes for approved clinical uses. Further, the technologies can be implemented to ensure compliance with privacy and/or health data residency requirements (e.g., U.S. Health Insurance Portability and Accountability Act, European General Data Protection Regulation, California Consumer Privacy Act, and the like).

[0365] Collaboration and sharing can be facilitated by policy-based access control for genomic digital data in any of a variety of workflows supporting the above as described herein. For example, tenants can collaborate on a workflow, results of a workflow can be passed from one tenant to another, and the like.

Example 47 – Example Use Cases: Primary, Secondary, and Tertiary Analyses

[0366] Sequencing generates large volumes of genomic digital data, and the analysis process associated with such data can be complex. A variety of analytical tools can be used to reveal meaningful information in the data in a timely manner. The technologies described herein can enable collaboration during use of the analytical tools and related workflows as well as providing the results of one workflow from one tenant to another. One way of describing the genomic digital data analysis process divides the analysis process into three main phases: primary, secondary, and tertiary data analysis. Some actions can be performed automatically on a sequencing instrument, while others occur after sequencing is completed.

[0367] Primary data analysis can include analysis performed during cycles of sequencing chemistry and imaging, which provides base calls and associated quality scores representing the primary structure of nucleotide strands. In one example, the output of primary data analysis are BCL base

call files indicating the base calls of clusters of nucleotide strands. In practice, such analysis can be performed automatically on sequencing systems. The results of primary analysis can take the form of genomic digital data embodied in files and uploaded to the cloud for further processing during secondary analysis. Collaboration and sharing can be facilitated by policy-based access control for such genomic digital data as described herein. For example, one tenant can perform primary analysis and provide access to the results to one or more tenants for secondary analysis.

[0368] Secondary analysis can take the results of primary analysis, which represent base calls of non-aligned nucleotide fragments, and provide a determination of full sequence or sequence range (e.g., a gene) by analysis and alignment of base calls of nucleotide fragments for a sample, from which genetic variants can be determined. For example, the output of secondary analysis can be in the form of FASTQ files that include sequence information and a quality score. Such analysis typically involves aligning and assembly of the nucleotide fragments. Given the full sequence or a sequence range, variants can be determined. Sequence alignment, variant calling, data visualization, RNA sequencing experiments, gene fusion detection, total RNA expression profiling, and determination of methylated bases can also be performed. Collaboration and sharing of genomic data during secondary analysis can be facilitated by policy-based access control for genomic digital data as described herein. For example, one tenant can perform secondary analysis and provide access to the results to one or more tenants for tertiary analysis.

[0369] Tertiary data analysis can include using any of a wide variety of biological data mining and interpretation tools on sequence data to convert data into knowledge. For example, variant interpretation and diagnosis can be performed on results of secondary analysis. Collaboration and sharing of genomic data during tertiary analysis can be facilitated by policy-based access control for genomic digital data as described herein. For example, tertiary data analysis can include a recommendation on whether the genomic data indicates that a patient will respond to a certain medical therapy (e.g., medicine, radiation, or the like).

Example 48 – Example Use Cases: Intra-Analysis Collaboration

[0370] In any of the examples herein, the policy-based access control technologies can be used for intra-analysis collaboration, where more than one party (e.g., tenant, workgroup, or both) collaborates to perform analysis within a phase.

Example 49 – Example Use Cases: Cross-Analysis Collaboration

[0371] In any of the examples herein, the policy-based access control technologies can be used for intra-analysis collaboration, where one or more parties (e.g., tenant, workgroup, or both) perform analysis that is then provided to one or more other parties to perform subsequent analysis in a different phase.

[0372] In such a case, a feedback loop of tertiary analysis results can be provided back to the party that performed the secondary analysis for revisions to re-running of the secondary analysis. The secondary analysis results can then be updated so that the tertiary analysis is revised or re-run (e.g., by the same or one or more other parties).

**Example 50 – Example Use Cases: Government Agency Approved
Instruments and Tests**

[0373] In any of the examples here, the policy-based access control technologies can be used to implement diagnostic processing across tenants for government-agency-approved diagnostic instruments and/or tests. For example, FDA approved instruments and/or tests can be conducted in scenarios where multiple tenants are involved and share data as part of a test.

Example 51 – Example Use Cases: Research Processing

[0374] Access control as described herein can enforce research use only processing. For example, research use only can be conducted by tenants or workgroups within tenants collaborating across institutional and geographical boundaries in a genomic digital data sharing scenario while preserving security of data. For example, access to individual patient identifiers can be restricted so that processing of data cannot be correlated to a specific individual.

Example 52 – Example Use Cases: Privacy and Data Residency

[0375] Further, access control can be implemented to ensure compliance with privacy and/or health data residency (e.g., geographical location) requirements. For example, in a research scenario, individual health data with directly identifying information can be blocked or restricted while aggregated health data sets with identifying information can be published or pushed to a third-party provider or other tenant for analysis.

[0376] In a diagnostic scenario, individual health data with directly identifying information can be permitted.

[0377] For example, an access token can be used to ensure other parties (third-party analysis providers) no longer have access to the shared data when the token expires or is revoked.

Revocation can be performed due to conclusion of processing or by patient instructions.

[0378] For example, an access token can be used to ensure that data resides in a certain geographic location or region.

Example 53 – Example Advantages

[0379] The policy-based sharing technologies can result in a number of advantages. For example, the ease with which sharing can be accomplished in a policy-based sharing environment can generally encourage sharing between tenants. Due to the late-binding nature of the role identifiers, it is not necessary to store a comprehensive mapping of users or tenants to roles. Instead, roles can be bound at execution time. Thus, the overall storage requirements for security data are reduced.

[0380] Similarly, the flexibility of policy-based role assignment allows incorporation of new criteria without having to redesign the platform or complicate the administration by the tenant.

[0381] Binding roles at execution time also provides more accurate role assignment. For example, a change in status or service level of a tenant can be immediately reflected rather than after some period of time in which pre-mapped roles are re-assigned.

[0382] Another advantage is that executable workflows can be shared along with the underlying data on which such workflows are executed. Thus, a tenant can share underlying data, execute a shared workflow on such underlying data, and receive analysis results. The workflow can further invoke external service providers, leading to a comprehensive collaboration scenario that is not possible without such technologies.

[0383] The trust relationships can be enforced via signed tokens as described herein. Thus, the security of the underlying data can be assured, enabling tenant-to-tenant sharing while preserving security of the underlying data. Auditing of access can also be achieved, and audit logs can be used for testing, security, or compliance purposes.

[0384] Software testing can also be more easily achieved by easily setting up a test tenant and sharing data with the test tenant, providing proof-of-concept and quality assurance testing for sharing scenarios that can then be extended to actual tenants outside of testing scenarios.

Example 54 – Example Temporary Cloud Provider Credentials via Secure Discovery Framework

[0385] FIG. 23 is an example system 2300 providing temporary cloud provider credentials via a secure discovery framework that can be implemented in any of the examples herein. In the example, a software-as-a-service (SaaS) platform at one or more data centers 2330 orchestrates access to genomic digital data resources via policy-based access control over the network 2305 as described herein. As shown, the system 2300 supports a plurality of tenants 2310 accessing genomic computing services (e.g., by the applications 2350). In the example, the tenants 2310 are identified by tenant identifiers and can comprise users identified by user identifiers 2320.

[0386] Although not shown, the system 2300 can comprise a policy store comprising policy-based access control definitions as described herein.

[0387] A platform authentication service 2335 comprises a cloud provider discovery service 2340 that can access a stored mapping between identities accessing the platform and cloud provider accounts of the cloud providers 2390A-N. An authorization checker 2345 can enforce the mapping to provide secure access as described herein.

[0388] A credentials management service can be used to derive limited temporary derived credentials from underlying credentials of the cloud provider accounts.

[0389] An administrative service 2358 is provided by which underlying credentials can be managed and policies applied to control access to limited temporary derived credentials. The administrative service 2358 can also manage the identity-to-cloud-provider-account mappings provided to the cloud provider discovery service 2340 and authorization checker 2345.

[0390] The discovery service 2340 can determine details of the cloud provider based on an incoming identity. Incoming requests for access to the cloud providers 2390 can be handled by the authorization checker 2345, which can access the mappings configured via the administrative service 2358.

[0391] Although the example shows applications 2350A that access the cloud providers 2390A-N on behalf of the user identifiers 2320, in practice, other identities can access the cloud providers 2390A-N as described herein.

[0392] Although not shown explicitly, the credentials management service 2380 can interact with the cloud providers 2390A-N to obtain credentials and provide credential management functions.

[0393] In practice, a genomic digital data resource can be linked to a role identifier and stored at a given cloud provider account. The limited temporary derived credentials can be used by an accessing identity to access the genomic data resource.

[0394] Although a single data center 2330 is shown, in practice, the platform can be distributed among a plurality of data centers having computing systems. Additional components can be included to implement security, redundancy, load balancing, report design, and the like.

Example 55 – Example Cloud Providers

[0395] In any of the examples herein, a variety of cloud service providers (or simply “cloud providers”) can be supported. In practice, a particular cloud provider account is associated with a provider type. Underlying credentials for a particular account can be stored and leveraged to generate limited temporary derived credentials for use by identities requesting access. Cloud resources such as storage can then be integrated into the platform as described herein.

[0396] Cloud providers provide their customers with accounts by which they can access resources (e.g., data, services, and the like) that are provided via hardware and software infrastructure maintained by the cloud provider. Different cloud providers can provide different arrangements, with different technical advantages, configuration options, licensing arrangements, and the like. In the examples herein, the resources can comprise genomic digital data resources such as the genomic digital data described herein.

[0397] The platform described herein can support a plurality of different cloud providers or provider types (e.g., Amazon Web Services, Microsoft Azure, Bitbucket, GitHub, GitLab, Google Cloud, and the like) per identity (e.g., per tenant). In this way, tenants can take advantage of their preferred cloud provider accounts while still availing themselves of the genomic computing services provided by the platform.

[0398] Such an arrangement is sometimes called “bring your own account” because the platform gives tenants the freedom to choose any supported external cloud provider. The described technologies can support one or more external cloud providers, one or more internal cloud providers (e.g., provided by a same provider providing the platform) or a mix of external and internal cloud providers.

Example 56 – Example Supported Identities

[0399] In any of the examples herein, any identity can take advantage of the temporary cloud provider credentials described herein. An identity can be one of a plurality of different identity types supported by the software-as-a-service platform. For example, supported identity types can comprise user, tenant, workgroup, application, project, or the like. An identity may have memberships that comprise other identities. For example, a user may be a member of a particular

tenant. Identities can be associated with a geographical region. For example, when an application is hosted, it can be associated with a home geographical region.

[0400] A workgroup identity can span a plurality of tenants of the software-as-a-service platform to facilitate cross-tenant access to cloud provider accounts.

[0401] The identity is typically indicated by an identifier that is unique within a scope.

[0402] Collaboration between users at different tenants can be achieved by implementing workgroups as described herein. Such cross-tenant workgroups can share access to a single cloud provider account as described herein.

Example 57 – Example Underlying Credentials

[0403] In any of the examples herein, a cloud provider account can be accessed via underlying credentials as described herein. Underlying credentials are typically root credentials that allow creation of further credentials on demand. As described herein, a credentials management service can implement the details of providing limited temporary derived credentials by leveraging the persisted underlying credentials. The logistics and process may differ for different cloud provider types, but still be transparent to the identity accessing the account because details are handled by the credentials management service.

[0404] In any of the examples herein, credentials can take forms such as usernames, passwords, tokens, signed digital certificates, and the like.

Example 58 – Example Limited Temporary Derived Credentials

[0405] In any of the examples herein, limited temporary derived credentials can be derived from underlying credentials to provide access to a cloud provider account. In practice, such derived credentials provide more limited access than the underlying (e.g., root) credentials. For example, a limited number of resources, limited rights (e.g., read only), or the like can be accessed instead of all possible access rights to the account. The derived credentials can also be made temporary (e.g., valid for 24 hours or the like). Upon expiration, a renewed credential can be obtained.

[0406] Thus, the level of granularity supported by the technologies described herein can vary, both from a rights perspective and from an identity perspective. On the rights perspective side, a different limited credentials can be provided to control access to resources (e.g., a read-only credential prevents writes by one identity, and a different, read-write credential allows writes by another identity). On the identity side, different credentials can be provided, even if they have the same limitations. For example, two different derived credentials can be provided to two different

identities, even if both credentials provide the same level of rights (e.g., read-only). An advantage of such a scenario is that if one identity is compromised, one of the credentials can be revoked while maintaining the other. And, the underlying credentials remain uncompromised because they need not be provided to accessing identities.

[0407] The bundle of rights provided in the derived credentials for a particular identity can be controlled by the policy-based access control technologies described herein (e.g., based on identity, identity memberships, or the like). Features such as regional tags can be supported to limit access to specified (geographical) regions.

[0408] Another use case is an Internet-of-things (IoT) scenario. Different derived credentials can be provided to different IoT devices so that risk is limited to a single device if it is compromised. In this way, access can be provided to a variety of identities while maintaining confidentiality of the underlying credentials.

[0409] Rights such as read, write, execute can thus be controlled and managed as desired.

[0410] In practice, derived credentials can take the form of a federated token that can be created and recognized by more than one authority (e.g., one authority creates in conjunction with authorization from another, trusted authority).

Example 59 – Example Credentials Management Service

[0411] In any of the examples herein, a credentials management service can be used to manage the underlying credentials of cloud provider accounts. Such a service can be internal to the software-as-a-service platform (e.g., as a microservice) or be implemented externally (e.g., as the Vault service, Centrify service, or the like).

[0412] The credentials management service can store underlying credentials and then generate limited temporary derived credentials as requested by an authorization checker (e.g., according to defined policies) as described herein.

Example 60 – Example Method of Providing Temporary Cloud Provider Credentials

[0413] FIG. 24 is a flowchart of an example overall method 2400 of providing temporary limited derived credentials.

[0414] At 2410, cloud provider accounts are managed; the management comprises storing 2420 underlying credentials for cloud provider accounts. As described herein, such underlying credentials can be used to create further (derived) credentials.

[0415] Subsequently, at 2430 the cloud provider can be accessed via limited temporary derived credentials derived from the underlying credentials. For example, genomic digital data can be read from and stored to a cloud provider account. In some cases, the cloud provider account can be provided by the cloud provider to a tenant of which a user identity is a member, but other identities can be supported. In this way, services can leverage the stored underlying credentials to integrate cloud provider accounts into the platform implementing the described technologies.

[0416] As a result, the platform can seamlessly work across cloud provider types transparently to the user while maintaining network security.

Example 61 – Example Method Managing Cloud Provider Accounts for Platform

[0417] FIG. 25 is a flowchart of an example method 2500 of managing cloud provider accounts to provide temporary limited derived credentials. In the example, at 2520, an administrative user identity accesses a credential manager to manage the cloud provider accounts within the platform.

[0418] At 2540, the credential manager receives underlying credentials for a cloud provider account. Other details of the cloud provider account can be received as described herein.

[0419] At 2550, the underlying credentials are persisted with a credentials management service. Persistence can comprise storing details of the account and storing a mapping between the account and an identity as described herein. Credential management policies can be stored and associated with the credentials and used to implement as rotation, the lease period, and the like.

Example 62 – Example Details of Managing Cloud Provider Accounts for Platform

[0420] FIG. 26 is a sequence diagram of an example implementation of a method 2600 of managing cloud provider accounts to provide temporary limited derived credentials. In the example, a tenant administrator identity accesses an administrative console (IAM) 2650 to configure and manage cloud provider credentials. On a basic level, the credentials of a particular cloud provider account can be provided to the administrative console 2650.

[0421] Administrators can come from levels other than the tenant. For example, a workgroup administrator can access the administrative console 2650. Administrators can be identified by a group in a hierarchy (e.g., platform.tenant.credmgmt, platform.workgroup.credmgmt, or the like); members of the group are permitted to manage the underlying credentials. As described herein, any number of cloud provider types can be supported, and details, including the type can be collected as part of management.

[0422] The administrative console 2650 can make a secure application programming interface (API) call to a platform authentication service 2660 to store the underlying credentials with the credentials management service 2670, which is subject to an authorization check 2665. The API call can include information indicating a type of the cloud provider (e.g., “/Credentials?type=<AWS>”) that can be used later for cloud provider discovery.

[0423] The credentials can then be persisted to a credentials management service 2670 along with credential management policies as described herein.

Example 63 – Example Credential Management Policies

[0424] In any of the examples herein, credential management policies can be stored with underlying credentials to configure how credentials are derived from the underlying credentials. Such policies can include rotation (e.g., to reduce credential lifetime), lease period (e.g., for how long a temporary credential is valid), and the like.

[0425] For example, derived credentials can be set to automatically expire after a limited time period (e.g., after 24 hours or the like).

Example 64 – Example Method of Providing Limited Temporary Derived Credentials

[0426] FIG. 27 is a flowchart of an example method 2700 of providing limited temporary derived credentials. Basically, the method 2700 leverages the underlying credentials provided via cloud provider account management described above to generate limited temporary derived credentials for use by identities accessing the cloud providers. The method 2700 can be performed in a computing system supporting a plurality of tenants accessing genomic computing services in a software-as-a-service platform that orchestrates access to genomic digital data resources via policy-based access control.

[0427] The method 2700 can be performed responsive to a particular request for access or set up in advance before access.

[0428] At some point, a request on behalf of an identity to grant access to resources of the cloud provider account is received. For example, an identity can seek access to resources at a cloud provider that are controlled by a cloud provider account that is mapped to the identity. In practice, an identity can launch a task or application that involves access to cloud provider resources, and access is sought on behalf of the identity, or access can be sought using an identity of the task or application.

[0429] In the example, at 2740, cloud provider details (e.g., account) are discovered based on the identity accessing the software-as-a-service platform. For example, account identifier, cloud provider identifier, cloud provider type, cloud provider access information, and the like can be discovered via a microservice or secure API provided as part of a platform authentication service. As described herein, the software-as-a-service platform supports limited temporary credentials for a plurality of cloud provider types. In a resource-based approach, a resource identifier can be used (e.g., in concert with the identity) to determine which cloud service details are to be used.

[0430] A request for limited temporary derived credentials can be received. As described herein, such credentials can be derived via underlying credentials of a cloud service account and provide access to resources of the cloud service account. Such access can be limited according to role-based policies set up during configuration. For example, a requesting identity can be associated with a role that has limited rights. Although a single cloud service account can be indicated based on the requesting identity (e.g., via a mapping from identity to cloud service account), a resource-based approach can be used instead or in addition to an identity-based approach. For example, a resource identifier can be included as part of the request, and a mapping of resource identifiers to cloud service accounts maintained. A discovery framework can thus determine which cloud service details are to be used based on the resource identifier specified in the request.

[0431] At 2750, the request for limited temporary derived credentials valid for the cloud provider account is sent (e.g., delegated) to a credentials management service. The request can include the permitted level of access (e.g., which can be limited to those rights permitted by policies configured to enforce policy-based access control). Policy-based access control can thus serve as a flexible infrastructure by which access can be controlled in an environment that supports the limited temporary derived credentials described herein while maintaining restrictions specified by policies that can be evaluated (e.g., at runtime).

[0432] In response, at 2760, limited temporary derived credentials valid for the cloud provider account are received from the credentials management service based on credentials management policies set during configuration. Such credentials are valid for the cloud provider account and provide the permitted level of access as configured. Options such as rotation and lease period can be implemented by the credentials management service as configured. The credentials can be limited to those rights permitted by policies configured to enforce policy-based access control. As described herein, the credentials can be derived from underlying credentials that were previously received for the cloud provider account.

[0433] At 2770, the limited temporary derived credentials valid for the cloud provider account are provided for use by the accessing identity.

[0434] At 2780, the resources of the cloud provider account are accessed via the limited temporary derived credentials (e.g., by or on behalf of the accessing identity). Such access can be limited according to the policy-based access control described herein. As noted herein, the credentials are typically provided on a temporary basis and eventually expire. Renewed credentials can be obtained by re-authenticating.

[0435] As described herein, the software-as-a-service platform can support multiple different cloud providers per single tenant. Also, the platform can support multiple different cloud provider accounts per a single tenant.

Example 65 – Example Resource Granularity Implementation

[0436] In any of the examples herein, a resource identifier can be provided as part of a request for limited temporary derived credentials. A mapping between resource identifiers and cloud provider accounts can be maintained. The resource identifier can then be mapped to the cloud provider account, and limited temporary derived credentials for the cloud provider account can be provided in response to the request. In such an implementation, a plurality of different cloud provider accounts (and/or cloud provider types) can be supported per a single tenant, depending on the resource.

Example 66 – Example Details of Providing Limited Temporary Derived Credentials

[0437] FIG. 28 is a sequence diagram of an example implementation of a method 2800 of providing limited temporary derived credentials. The method 2800 can be used by any of a variety of services to leverage the cloud credential technologies described herein.

[0438] In the example, a user identity 2810 is accessing a software-as-a-service platform that includes a genomics analysis service 2850 and a platform authentication service 2860. In the example, the credentials management service 2870 is external to the platform but can alternatively be implemented as internal to the platform.

[0439] As part of launching a workflow execution service, the user identity 2810 can provide a signed access token 2815 (e.g., JSON Web Token or the like) that indicates memberships such as workgroup, tenant, proxy tenant (e.g., external third party provider), project, or the like. In the example, an analysis service (e.g., Illumina Analysis Pipeline or other service) receives the token

2815 and proceeds to send a request for cloud service details to the discovery service 2865 of the platform authentication service 2860.

[0440] The discovery service 2865 can provide cloud provider details 2855 in response. Such details 2855 can comprise the cloud provider type, account name, connection details, and the like. The details 2855 can be based on memberships of the user identity 2810 such as tenant, workgroup, or the like. For example, the discovery service 2865 can consult a mapping between a tenant and a cloud provider account to determine that the tenant uses a particular cloud provider account.

However, as described herein, finer granularity can be provided (e.g., a mapping from workgroup or even user to a particular cloud provider account). Thus per-tenant, per-workgroup, or per-user cloud provider account configuration can be supported. Different cloud provider account types can be supported and cross-tenant scenarios can be supported (e.g., a workgroup has members of two different tenants, and members of the workgroup have access to a same cloud provider account).

[0441] The service 2850 can then request credentials from an authorization checker 2868 that verifies whether access is allowed and then delegates the request to a credentials management service 2870 according to role-based policies that indicate what level of access is permitted.

[0442] The credentials management service 2870 then generates the limited temporary derived credentials 2875 as described herein. In practice, the credentials management service 2870 can work with the cloud provider 2890 to generate limited temporary derived credentials according to policies configured at the credentials management service 2870. Features such as rotation, lease, and the like can be supported according to policies configured for the credentials management service 2870. Such credentials 2875 are then passed back to the platform authentication service 2860, which relays them to the requesting service 2850.

[0443] With the limited temporary derived credentials 2875, the service 2850 can then access data or other resources at the cloud provider 2890.

[0444] Although the example shows a request by a user identity, in practice, identities such as tenants, workgroups, projects, or applications can also be supported as described herein.

Example 67 – Example Cloud Provider Mapping

[0445] FIG. 29 is a block diagram of an example cloud provider mapping 2900. In the example, identities are mapped to cloud providers 2990W-Z, which can be of different cloud provider types. Although not shown, the mapping can provide further details, such as a particular account on the cloud provider.

[0446] As shown, a particular tenant identity 2910N can be mapped to a particular cloud provider 2990Z. An application identity 2920 can be mapped to a particular cloud provider 2990X.

[0447] The level of granularity can be at the workgroup level, so that different workgroups 2915J, 2915K of same tenant 2910A are mapped to different cloud providers 2990W, 2990X. Further, a cross-tenant situation can be supported so that a cross-tenant workgroup identity 2915K is mapped to a cloud provider 2990W.

[0448] Other identities can be mapped such as users, projects, or proxy tenants (e.g., external third-party tenants as described herein).

[0449] In practice, the mapping 2900 can be implemented as a table that associates identifiers with cloud providers. Precedence can be observed so that a workgroup mapping overrides a default tenant mapping or the like.

[0450] Such a mapping can be used during discovery to determine the cloud provider and further details for a given identity requesting access.

Example 68 – Example User Interface

[0451] FIG. 30 is a screen shot 3000 of an example user interface of a credentials management tool providing limited temporary derived credentials management. In the example, a window 3050 allows an administrative user to provide cloud provider details for a particular tenant 3015A. As shown, a cloud provider type, cloud provider name, access key, and secret key are supported. In practice, different arrangements can be provided and can differ by cloud provider type.

[0452] By convention, the default information for a particular tenant is shown. However, different provider information can be provided per workgroup (e.g., using the Add Workgroup 3015B button).

Example 69 – Example Credentials Object

[0453] FIG. 31 is a block diagram of an example credentials object 3100 for underlying credentials. Such an object 3100 can be used to persist credentials information at the platform authentication service. In the example, the credentials object 3100 comprises a cloud provider type that has a value indicating a type of cloud provider for the credentials, a credentials name 3120 that allows an instance of the credentials object 3100 to have a unique (e.g., user friendly) identifier, and an identity cluster 3130. The identity cluster 3130 can comprise both an identity type and an identity identifier. For example, the identity type can indicate tenant, workgroup, project, application, user, or the like, and the identity identifier can identify the identity uniquely.

[0454] The underlying credentials storage 3150 can store the underlying credentials, such as an access key 3155, secret key 3157, or the like. In practice, the storage 3150 can be implemented as a JSON object.

[0455] In a simple implementation, the credentials object can be persisted per tenant to provide different cloud provider accounts for different tenants. However, finer grained granularity can be implemented that allows plural objects 3100 per tenant (e.g., per workgroup, user, or the like). As described herein, an application can have its own credentials object 3100.

[0456] An example credentials object 3100 can take the following form:

[0457] {

[0458] “type”: “aws”,

[0459] “name”: “tenant01-aws-provider”,

[0460] “secret”: {“access_key”: “accessKey01”, “secret_key”; “secret”},

[0461] “identity” : “tid:tenantID”}

[0462] Where,

[0463] type = type of credential

[0464] name = user defined credential name for easy lookup

[0465] Secret = JSON object which has actual credential information

[0466] identity format = tid:tenantID, wid:workgroupId, uid:userId, aid:appID

Identity	Description
tid	Tenant Identity ID managed by Platform
wid	Workgroup Identity ID managed by Platform
uid	User Identity ID managed by Platform
aip	Application Identity ID managed by Platform
cid	Context Identity ID managed by Platform

Example 70 – Example Use Case: Genomic Analysis Pipeline

[0467] In any of the examples herein, an example use case is a genomic analysis pipeline application. In practice, a set of run parameters are submitted to the genomic analysis pipeline. Federation layers can use global events that are understood across platforms. A basic use case is to look at identity information associated with the run parameters. So, for example, workgroup identity information associated with the run can be used to find the cloud provider account and

generate limited temporary derived credentials derived from root credentials of the account associated with the workgroup and return for such credentials for use by the pipeline orchestrator to access the cloud provider account associated with the workgroup.

[0468] In practice, an application can launch the genomic analysis pipeline by supplying parameters to the pipeline, and such parameters may themselves be inputs to the application. An example parameter is a resource name. The application then requests credentials for accessing the resource specified in input parameters. The resource can be a file located in some geographical location. When the application requests credentials for the resource, the discovery framework can provide the limited temporary derived credentials as described herein by which the resource can be accessed. Such credentials are provided only if authorization is indicated at the time of request (e.g., by policy-based access control). Regional restrictions can be imposed as described herein.

[0469] In any of the examples herein, a request for credentials can include a resource identifier. Therefore, different resources can be mapped to different cloud service accounts, and a request can be processed accordingly.

[0470] A basic implementation can map a workgroup to a single cloud service account and a single region. Multiple regions can add complexities of legal and organizational compliance.

Example 71 – Example Use Case: Hybrid Data Sources

[0471] In any of the examples herein, genomic analysis can proceed with digital resources from both cloud provider accounts and local (e.g., on-premises) accounts.

[0472] Hybrid regions can also be supported so that genomic analysis can proceed with digital resources maintained in a region different from where the request originated or from two different regions. Regional restrictions can be put into place to prevent access to unauthorized or impermissible regions.

Example 72 – Example Identity Hierarchy

[0473] FIG. 32 is a block diagram of an example identity hierarchy 3200. At the top of the hierarchy is the platform global admin console identity 3210 which can be associated with global administrators 3215K. The global admin console identity 3210 can perform the widest variety of actions, including creating new identities.

[0474] The tenant identity 3220A-B is associated with tenant administrators 3225A-B and can perform the widest variety of actions for a particular tenant, including creating new identities within the tenant (e.g., workgroups, users, etc.).

[0475] The next level is the workgroup identity 3230A-N which is associated with workgroup administrators 3235A-N, who can perform actions for a particular workgroup.

[0476] Then, there are user identities 3240A-N.

[0477] Separately, there can be platform identities 3290 that can include a customer-facing application 3292A-B, an internal application 3297, or the like. Such identities can be used by applications when accessing cloud providers (e.g., regardless of the identity launching the application).

[0478] In practice, a particular user can be an administrator at the workgroup, tenant, or global level. In practice, special protections can be put in place for global admins 3215K (or certain ones of them) so that they do not have direct access to data of particular tenants.

[0479] The following hierarchy can be implemented to control access to credentials features:

Identity	Create	List	Read	Rotate	Destroy
Platform Global Admin (PGA)	Yes – For storing infrastructure credentials	Yes – All	Yes (e.g., only the infra- structure credentials persisted by PGA admin)	Yes – All	Yes – All
Application	Create credentials at the corresponding app level and across tenants, workgroups and user	Yes – Restricted to app permissions	Yes – Restricted to app permissions	Yes – Restricted to app permissions	Yes – Restricted to app permissions
Tenant Admin	Yes – (Tenant, WG level)	Yes – (Tenant, WG level)	Yes – (Tenant, WG level)	Yes – (Tenant, WG level)	Yes – (Tenant, WG level)
Workgroup Admin	Yes – (WG level)	Yes – (WG level)	Yes – (WG level)	Yes – (WG level)	Yes – (WG level)
User	Yes – (User Level)	Yes – (User Level)	Yes – (User Level)	Yes – (User Level)	Yes – (User Level)

Example 73 – Example Integration with Policies

[0480] In any of the examples herein the limited temporary derived credentials features described herein can be integrated with the policy-based access control features described herein. For example, policies can be created to control access to resource identifiers that identify resources stored at a cloud provider.

[0481] From a user standpoint, the fact that the resource is stored at a cloud provider can be transparent. The same functionality and same authentication process can be supported. However, if special authentication is required by the cloud provider (e.g., two-factor authentication), it can be supported differently according to the conventions of a particular cloud provider.

[0482] Using policy-based access control, a policy can be attached to any identity described herein, including applications. The policy-based access control can serve as a base layer on top of which the temporary credentials technology operates. For example, a policy can be set up for a workgroup and specify that the workgroup can access data only in certain geographical regions. If the workgroup is provided with applications, such applications will be restricted by the policy but can themselves have additional restrictions based on the identity of the application. Requests for credentials will be limited to those restrictions specified by policy. The policy-based access control infrastructure thus provides a flexible authorization layer based on customizable policies that can work in concert with the temporary credentials described herein. Any additional logic can be applied via policy (e.g., access only during certain times of day, access from a particular range of IP addresses, or the like).

[0483] The discovery framework can thus apply the authorization layer provided by policy-based access control before it returns the limited temporary derived credentials.

[0484] However, any of the limited temporary derived credentials features described herein can be implemented independently of (e.g., without implementing) the policy-based access control features described herein. Policy-based access control is therefore not an essential feature of the limited temporary derived credentials technologies described herein. For example, an authorization layer that is not policy driven can work in concert with the discovery framework. A simple mapping of whether an identity has access to a region can be applied instead of a policy as described herein.

Example 74 – Example Advantages

[0485] The limited temporary derived credentials features described herein can provide advantages from a technical perspective in that fine granularity is provided to tenants who wish to use a plurality of different cloud provider accounts or cloud provider types. Also, different credentials (e.g., credentials instances) can be provided to different identities for the same cloud provider account without intervention by administrators, increasing overall network security.

[0486] Cross-tenant workgroups can be supported so that collaboration between groups at different tenants can be achieved, regardless of the cloud provider involved.

[0487] From a user perspective, the fact that data is being stored at a cloud provider account can be transparent in that the same features and authentication process can be supported across different cloud provider types.

[0488] Other advantages include improved administration of credentials and integration into a policy-based access control environment.

Example 75 – Example Computing Systems

[0489] FIG. 33 depicts an example of a suitable computing system 3300 in which digital aspects of the described innovations can be implemented. The computing system 3300 is not intended to suggest any limitation as to scope of use or functionality of the present disclosure, as the innovations can be implemented in diverse computing systems.

[0490] With reference to FIG. 33, the computing system 3300 includes one or more processing units 3310, 3315 and memory 3320, 3325. In FIG. 33, this basic configuration 3330 is included within a dashed line. The one or more processing units execute computer-executable instructions, such as for implementing the features described in the examples herein. The one or more processing units 3310, 3315 can be any combination or central processing units (CPUs), graphical processing units (GPUs), single core processors, multi-core processors, application-specific integrated circuits (ASICs), programmable circuits such as Field Programmable Gate Arrays (FPGA), and the like. One or more of the processing unit(s) 3310, 3315 may be implemented in software (e.g., ultimately executed on hardware) and/or firmware in addition to hardware implementations.

[0491] In a multi-processing system, multiple processing units execute computer-executable instructions to increase processing power. The tangible memory 3320, 3325 can be volatile memory (e.g., registers, cache, RAM), non-volatile memory (e.g., ROM, EEPROM, flash memory, etc.), or some combination of the two, accessible by the processing unit(s) 3310, 3315. The memory 3320, 3325 stores software 3380 implementing one or more innovations described herein, in the form of computer-executable instructions suitable for execution by the processing unit(s) 3310, 3315.

[0492] Functionality can also be performed, at least in part, by one or more hardware logic components. For example, Field-programmable Gate Arrays (FPGAs), Application-specific Standard Products (ASSPs), System-on-a chip systems (SOCs), Complex Programmable Logic Devices (CPLDs), and the like can be used.

[0493] A computing system 3300 can have additional features. For example, the computing system 3300 includes storage 3340, one or more input devices 3350, one or more output devices 3360, and one or more communication connections 3370, including input devices, output devices, and communication connections for interacting with a user. An interconnection mechanism (not shown) such as a bus, controller, or network interconnects the components of the computing system 3300. Typically, operating system software (not shown) provides an operating environment for other software executing in the computing system 3300, and coordinates activities of the components of the computing system 3300.

[0494] The tangible storage 3340 can be removable or non-removable, and includes magnetic disks, magnetic tapes or cassettes, CD-ROMs, DVDs, or any other medium which can be used to store information in a non-transitory way and which can be accessed within the computing system 3300. The storage 3340 stores instructions for the software 3380 implementing one or more innovations described herein.

[0495] The input device(s) 3350 can be an input device such as a keyboard, mouse, pen, or trackball, a voice input device, a scanning device, touch device (e.g., touchpad, display, or the like) or another device that provides input to the computing system 3300. The output device(s) 3360 can be a display, printer, speaker, CD-writer, or another device that provides output from the computing system 3300.

[0496] The communication connection(s) 3370 enable communication over a communication medium to another computing entity. The communication medium conveys information such as computer-executable instructions, audio or video input or output, or other data in a modulated data signal. A modulated data signal is a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media can use an electrical, optical, RF, or other carrier.

[0497] The innovations can be described in the context of computer-executable instructions, such as those included in program modules, being executed in a computing system on a target real or virtual processor (e.g., which is ultimately executed on one or more hardware processors). Generally, program modules or components include routines, programs, libraries, objects, classes, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The functionality of the program modules can be combined or split between program modules as desired in various embodiments. Computer-executable instructions for program modules can be executed within a local or distributed computing system.

[0498] For the sake of presentation, the detailed description uses terms like “determine” and “use” to describe computer operations in a computing system. These terms are high-level descriptions for operations performed by a computer and should not be confused with acts performed by a human being. The actual computer operations corresponding to these terms vary depending on implementation.

Example 76 – Computer-Readable Media

[0499] Any of the computer-readable media herein can be non-transitory (e.g., volatile memory such as DRAM or SRAM, nonvolatile memory such as magnetic storage, optical storage, or the like) and/or tangible. Any of the storing actions described herein can be implemented by storing in one or more computer-readable media (e.g., computer-readable storage media or other tangible media). Any of the things (e.g., data created and used during implementation) described as stored can be stored in one or more computer-readable media (e.g., computer-readable storage media or other tangible media). Computer-readable media can be limited to implementations not consisting of a signal.

[0500] Any of the methods described herein can be implemented by computer-executable instructions in (e.g., stored on, encoded on, or the like) one or more computer-readable media (e.g., computer-readable storage media or other tangible media) or one or more computer-readable storage devices (e.g., memory, magnetic storage, optical storage, or the like). Such instructions can cause a computing system to perform the method. The technologies described herein can be implemented in a variety of programming languages.

Example 77 – Example Cloud Computing Environment

[0501] FIG. 34 depicts an example cloud computing environment 3400 in which the described technologies can be implemented, including, e.g., the system 100 of FIG. 1 and other systems herein. The cloud computing environment 3400 comprises cloud computing services 3410. The cloud computing services 3410 can comprise various types of cloud computing resources, such as computer servers, data storage repositories, networking resources, etc. The cloud computing services 3410 can be centrally located (e.g., provided by a data center of a business or organization) or distributed (e.g., provided by various computing resources located at different locations, such as different data centers and/or located in different cities or countries).

[0502] The cloud computing services 3410 are utilized by various types of computing devices (e.g., client computing devices), such as computing devices 3420, 3422, and 3424. For example, the

computing devices (e.g., 3420, 3422, and 3424) can be computers (e.g., desktop or laptop computers), mobile devices (e.g., tablet computers or smart phones), or other types of computing devices. For example, the computing devices (e.g., 3420, 3422, and 3424) can utilize the cloud computing services 3410 to perform computing operations (e.g., data processing, data storage, and the like).

[0503] In practice, cloud-based, on-premises-based, or hybrid scenarios can be supported.

Example 78 – Example Implementations

[0504] Although the operations of some of the disclosed methods are described in a particular, sequential order for convenient presentation, such manner of description encompasses rearrangement, unless a particular ordering is required by specific language set forth herein. For example, operations described sequentially can in some cases be rearranged or performed concurrently.

Example 79 – Example Embodiments for Policy-Based Access Control

[0505] Any of the following embodiments can be implemented.

[0506] Clause 1. A method comprising:

[0507] in a computing system comprising a plurality of tenants seeking access to genomic digital data resources provided by one or more genomic data services in a software-as-a-service platform that orchestrates access to the genomic digital data resources via policy-based access control, receiving a policy-based access control definition for a first of the tenants for a given genomic digital data resource;

[0508] receiving a request for access to the given genomic digital data resource from a second of the tenants seeking access to the given genomic digital data resource; and

[0509] for the second of the tenants, granting access to the given genomic digital data resource based on the policy-based access control definition.

[0510] Clause 2. The method of Clause 1 wherein:

[0511] access to the given genomic digital data resource is controlled by a role identifier linked to the policy-based access control definition; and

[0512] the method further comprises:

[0513] responsive to the request for access, providing the role identifier specified in the policy-based access control definition for the request for access.

[0514] Clause 3. The method of Clause 2 wherein:

[0515] assigning the role identifier comprises late binding of the role identifier to a user identifier or tenant identifier of the request for access.

[0516] Clause 4. The method of any one of Clauses 2-3 further comprising:

[0517] responsive to the request for access, generating a signed access token containing the role identifier;

[0518] wherein access is granted based on presence of the role identifier in the signed access token.

[0519] Clause 5. The method of Clause 4 wherein:

[0520] access is further granted based on validation of the signed access token.

[0521] Clause 6. The method of any one of Clauses 4-5 further comprising:

[0522] publishing a signed grant token comprising the role identifier and a tenant identifier of a role administrator of the role identifier;

[0523] wherein access is further granted based on whether the tenant identifier of the signed grant token has sufficient rights to grant resources specified for the role identifier.

[0524] Clause 7. The method of any one of Clauses 1-6, wherein the first of the tenants comprises a proxy tenant representing an external service provider for which policy-based sharing is implemented.

[0525] Clause 8. The method of any one of Clauses 1-7 wherein the policy-based access control definition comprises a reference to a smart contract.

[0526] Clause 9. The method of any one of Clauses 1-8 wherein the policy-based access control definition comprises a reference to a service level of the second of the tenants, and access is granted according to the service level of the second of the tenants determined at a time of the request.

[0527] Clause 10. The method of any one of Clauses 1-9 wherein the policy-based access control definition specifies one or more access control statements comprising a filter attribute and a filter parameter.

[0528] Clause 11. The method of Clause 10 wherein the filter parameter specifies a wildcard for the filter attribute.

[0529] Clause 12. The method of any one of Clauses 10-11 wherein the filter attribute comprises an application.

[0530] Clause 13. The method of any one of Clauses 10-12 wherein the filter attribute comprises an application role identifier.

[0531] Clause 14. The method of any one of Clauses 1-13 wherein the policy-based access control definition supports access control statements that specify an access outcome, a tenant identifier, and one or more conditions under which access is granted.

[0532] Clause 15. The method of any one of Clauses 1-14 wherein the policy-based access control definition supports public access, private access, and application-based access.

[0533] Clause 16. The method of any one of Clauses 1-15 wherein the policy-based access control definition comprises a parameter evaluated at execution time.

[0534] Clause 17. The method of Clause 16 wherein:

[0535] the parameter of the policy-based access control definition comprises an application identifier parameter; and

[0536] granting access comprises comparing the application identifier parameter of the policy-based access control definition with an application identifier specified by the second of the tenants seeking access to the genomic digital data resources.

[0537] Clause 18. The method of any one of Clauses 16-17 wherein:

[0538] the parameter of the policy-based access control definition comprises a tenant identifier parameter; and

[0539] granting access comprises comparing the tenant identifier parameter of the access control definition with a tenant identifier of the second tenant seeking access to the genomic digital data resources.

[0540] Clause 19. A multi-tenant, cloud-based system comprising:

[0541] one or more processors;

[0542] memory coupled to the one or more processors;

[0543] a policy store comprising a policy-based access control definition received for a first tenant and comprising a role identifier;

[0544] a genomic digital data resource linked to the role identifier;

[0545] wherein the memory comprises computer-executable instructions causing the one or more processors to perform operations comprising:

[0546] receiving a request for access to the genomic digital data resource from a second tenant seeking access to the genomic digital data resource; and

[0547] for the second tenant, granting access to the genomic digital data resource according to the policy-based access control definition evaluated at a time of the request for access.

[0548] Clause 20. One or more computer-readable media comprising:

[0549] computer-executable instructions capable of causing a computing system to receive a publishing request for a first tenant to provide access to genomic digital data, wherein access to the

genomic digital data is controlled by a role identifier linked to a policy document, wherein the policy document comprises one or more conditions;

[0550] computer-executable instructions capable of causing a computing system to receive a request from a second tenant for access to the genomic digital data to which access is controlled by the role identifier linked to the policy document, wherein the request comprises one or more attributes;

[0551] computer-executable instructions capable of causing the computing system to access the policy document responsive to the request from the second tenant for access; and

[0552] computer-executable instructions capable of causing the computing system to generate an access token based on the one or more attributes and the one or more conditions, wherein the role identifier is included in the access token responsive to determining that the one or more conditions are fulfilled by the one or more attributes. and the access token authorizes access to the genomic digital data via the role identifier.

[0553] Clause 21 One or more computer-readable media comprising computer-executable instructions that when executed by a computing system, cause the computing system to perform the method of any one of Clauses 1-18.

Example 80 – Example Embodiments

[0554] Any of the following embodiments can be implemented.

[0555] Clause 1. A computer-implemented method comprising:

in a computing system supporting a plurality of tenants accessing genomic computing services in a software-as-a-service platform that orchestrates access to genomic digital data resources via policy-based access control,

discovering a cloud provider account for an identity accessing the software-as-a-service platform;

sending a request to a credentials management service for limited temporary derived credentials valid for the cloud provider account;

receiving the limited temporary derived credentials valid for the cloud provider account;

and

providing the limited temporary derived credentials for use by the identity.

Clause 2. The computer-implemented method of Clause 1 wherein:
the software-as-a-service platform supports multiple different cloud provider account types per a single tenant.

Clause 3. The computer-implemented method of any one of Clauses 1-2 wherein:
the software-as-a-service platform supports multiple different cloud provider accounts per a single tenant.

Clause 4. The computer-implemented method of any one of Clauses 1-3 further comprising:
receiving policy-based access control configuration information for the plurality of tenants;
wherein the limited temporary derived credentials are limited to rights indicated in the policy-based access control configuration information.

Clause 5. The computer-implemented method of any one of Clauses 1-4 further comprising:
receiving underlying credentials for the cloud provider account;
wherein the limited temporary derived credentials are derived from the underlying credentials.

Clause 6. The computer-implemented method of any one of Clauses 1-5 wherein:
the software-as-a-service platform supports limited temporary derived credentials for a plurality of cloud provider types.

Clause 7. The computer-implemented method of Clause 6 wherein:
discovering the cloud provider account comprises discovering a cloud provider type of the cloud provider account.

Clause 8. The computer-implemented method of Clause 7 wherein:
the credentials management service is external to the software-as-a-service platform.

Clause 9. The computer-implemented method of any one of Clauses 1-8 wherein:

the identity is one of a plurality of different identity types supported by the software-as-a-service platform.

Clause 10. The computer-implemented method of Clause 9 wherein:
the identity is of type “application.”

Clause 11. The computer-implemented method of any one of Clauses 9-10 wherein:
the identity is of type “workgroup.”

Clause 12. The computer-implemented method of Clause 11 wherein:
a workgroup identity spans a plurality of the tenants of the software-as-a-service platform.

Clause 13. The computer-implemented method of any one of Clauses 9-12 wherein:
the identity types supported by the software-as-a-service platform comprise:
application;
tenant;
workgroup; and
user.

Clause 14. The method of any one of Clauses 1-13 wherein:
the cloud provider account stores a genomic digital data resource;
access to the genomic digital data resource is controlled by a role identifier linked to a
policy-based access control definition; and
the method further comprises:
responsive to a request for access to the genomic digital data resource, providing the role
identifier specified in the policy-based access control definition for the request for access.

Clause 15. A multi-tenant, cloud-based system comprising:
one or more processors;
memory coupled to the one or more processors;
a mapping between identities accessing a software-as-a-service platform and cloud provider
accounts;
a policy store comprising policy-based access control definitions;

a genomic digital data resource linked to a role identifier and stored at a given cloud provider account external to the software-as-a-service platform;

wherein the memory comprises computer-executable instructions causing the one or more processors to perform operations comprising:

based on the mapping, discovering a cloud provider account for an identity accessing the software-as-a-service platform;

sending a request to a credentials management service for limited temporary derived credentials valid for the cloud provider account;

receiving the limited temporary derived credentials valid for the cloud provider account;

and

providing the limited temporary derived credentials for use by the identity to access the genomic digital data resource.

Clause 16. The system of Clause 15 wherein:

the software-as-a-service platform supports multiple different cloud provider account types per tenant; and

the software-as-a-service platform supports multiple different cloud provider accounts per tenant.

Clause 17. The system of any one of Clauses 15-16 wherein the memory further comprises computer-executable instructions causing the one or more processors to perform operations comprising:

granting access to the genomic digital data resource according to a policy-based access control definition evaluated at a time of a request for access.

Clause 18. The computer-implemented method of any one of Clauses 15-17 wherein:

the identity is one of a plurality of different identity types supported by the software-as-a-service platform.

Clause 19. The computer-implemented method of any one of Clauses 15-18 wherein:

the identity is of type “application.”

Clause 20. The computer-implemented method of any one of Clauses 15-18 wherein: the identity is of type “workgroup.”

Clause 21. The computer-implemented method of Clause 20 wherein: a workgroup identity spans a plurality of tenants of the software-as-a-service platform.

Clause 22. One or more computer-readable media comprising:
computer-executable instructions capable of causing a computing system to perform the following in a computing system supporting a plurality of tenants accessing genomic computing services in a software-as-a-service platform that orchestrates access to genomic digital data resources via policy-based access control:

discovering a cloud provider account for an identity accessing the software-as-a-service platform;

sending a request to a credentials management service for limited temporary derived credentials valid for the cloud provider account;

receiving the limited temporary derived credentials valid for the cloud provider account;
and

providing the limited temporary derived credentials for use by the identity to access the genomic digital data resources at the cloud provider account according to the policy-based access control;

wherein:

the software-as-a-service platform supports multiple different cloud provider account types per tenant; and

the software-as-a-service platform supports multiple different cloud provider accounts per tenant.

Clause 23. One or more computer-readable media comprising:
computer-executable instructions capable of causing a computing system to perform the method of any one of the Clauses 1-14.

Example 80 – Example Alternatives

[0556] The technologies from any example can be combined with the technologies described in any one or more of the other examples. In view of the many possible embodiments to which the principles of the disclosed technology can be applied, it should be recognized that the illustrated embodiments are examples of the disclosed technology and should not be taken as a limitation on the scope of the disclosed technology. Rather, the scope of the disclosed technology includes what is covered by the scope and spirit of the following claims.