Integrating and Accessing Medical Data Resources within the ViroLab Virtual Laboratory

Matthias Assel[1], Piotr Nowakowski[2], and Marian Bubak[2,3]

[1] High Performance Computing Center, University of Stuttgart, D-70550, Germany
[2] Academic Computer Centre CYFRONET AGH, ul. Nawojki 11, 30-950 Kraków, Poland
[3] Institute of Computer Science, AGH, al. Mickiewicza 30, 30-059, Kraków, Poland
assel@hlrs.de, p.nowakowski@cyfronet.pl, bubak@agh.edu.pl

Abstract. This paper presents the data access solutions which have been developed in the ViroLab Virtual Laboratory infrastructure to enable medical researchers and practitioners to conduct experiments in the area of HIV treatment. Such experiments require access to a number of geographically distributed data sets (residing at various hospitals) with heavy focus on integration and security issues. Scientists conducting virtual experiments need to be able to manipulate such distributed data in a consistent and secure manner. We describe the main components of the Virtual Laboratory framework being devoted to data access and explain how data is processed in the presented environment.

Keywords: The Grid, Data access, Virtual laboratories, Data integration, OGSA-DAI, Medical research

1 Introduction and Motivation

The ViroLab Virtual Laboratory is an integrated system of tools for accessing and integrating resources, whose main purpose is to facilitate medical research and treatment in the HIV virology domain as well as other types of research in the general field of medical sciences. The research is carried out in a collaborative working environment using state-of-the-art Grid computing technologies and standards [19] and consisting of distributed computing and data resources deployed at various networked sites. As the complexity of interfacing such resources often presents a steep learning curve for application developers, the main goal of the Virtual Laboratory (VL) is to present a powerful and flexible development environment for virtual experiment developers while preserving ease of use and reusability of the proposed solution, thus allowing transparent and secure access to corresponding underlying infrastructures. A detailed description of the ViroLab Virtual Laboratory design is outside the scope of this paper, but can be found in [21] and [22]. In this paper, we focus on aspects related to data retrieval, integration,, and manipulation in the VL environment.

M. Bubak et al. (Eds.): ICCS 2008, Part III, LNCS 5103, pp. 90–99, 2008.

As one can expect from a virtual laboratory for research and treatment in the area of medical sciences, the system must provide access to a range of medical data including genetic, treatment, and drug information. This data, used to conduct experiments in viral drug resistance interpretation and selecting optimal treatment strategies, comes from various hospitals and medical centers being partners in the ViroLab project and is secured against unauthorized access. From the point of view of experiment developers and users, all data records describing HIV subtypes and mutations are equally valuable and should be treated in an analogous manner. However, a problem emerges with achieving a uniform representation of such data (see next section for details). It is therefore the task of the data access component(s) of the ViroLab Virtual Laboratory to integrate data derived from various sources and to enable experiment developers to manipulate this data in a consistent, efficient and straightforward way.

Due to the sensitivity and confidentiality of data shared within the virtual laboratory, a very critical issue for developing services that allow access to distributed medical databases concerns the overall security including access control to certain resources (who is able to access which information set) as well as data encryption and integrity of relevant data sets processed by the data access infrastructure. ViroLab meets this important issue by introducing a highly dynamic and flexible environment that guarantees security on several levels using established security principles and technologies as described in [2], to protect the confidential information and to keep the patients privacy.

The remainder of this paper is structured as follows: Section 2 contains a description of related work and parallel projects where data access issues are covered. Sections 3 and 4 cover the integration and aggregation of sensitive medical data in the ViroLab project, while section 5 explains how such data can be manipulated by developers of experiments in the presented Virtual Laboratory. Section 6 presents areas of application of the presented technologies and section 7 contains conclusions and closing remarks.

2 Related Work

Data access in the Grid environments has been a subject of study and research for quite some time. Early solutions, such as those employed in batch Grid systems (for instance [16]) relied on replicating data contained in custom-tailored data repositories which were managed by Grid middleware. Prior to performing any calculations, data had to be fetched and staged by a specialized middleware component. Furthermore, when submitting a Grid job, the developer had to specify in advance which data elements (represented by files) were required for the computational task to proceed. Naturally, this was a limiting solution in that it took no notice of structured data storage technologies (such as databases) and did not provide for altering the available input data pool once the job was submitted for processing. Moreover, results had to be collected following the execution of the job and could not typically be stored on the fly as the job progressed. These constraints gave rise to a number of projects aiming at

standardization and increased flexibility of data access in Grids, the most important of them being OGSA-DAI [1]. The aim of this project is to develop a middleware system to assist with access and integration of data from separate sources via the Grid. The OGSA-DAI Toolkit supports the smooth exposure of various types of data sources such as relational or XML databases on to grids, and provides easy access to them through common Web Service interfaces. OGSA-DAI is successfully adopted in several research projects such as SIMDAT [20] and BeINGrid [17], and more.

We intend to follow up on the achievements of the aforementioned technologies and further adapt them to the needs of medical researchers in the ViroLab environment. It should be noted that a number of other Grid research initiatives exist, which deal with similar data management issues. Of note is the EUResist project [15], which aims at developing a European integrated system for clinical management of antiretroviral drug resistance. Peer-to-peer data storage solutions are being investigated in specific contexts, such as the the SIGMCC framework [8] or GREDIA [6]. Further discussion on data access and integration solutions applied in medical Grids can be found in both [14] and [7] while similar cluster solutions are discussed in [10]. However, while such technologies are aimed at a narrow group of applications, the solution developed in ViroLab is more generic in nature and allows experiment developers to securely access integrated medical data sets as well as ad-hoc databases used for the purposes of specific experiments instead of being focused on a single type of application or use case.

3 Integration of Heterogeneous (Bio)Medical Data Resources into the Laboratory Infrastructure

Accessing a local database is one of the most common and well-known procedures today but dealing with multiple and distributed systems simultaneously still implies lots of integrational work and results quite often in a real challenge for both administrators and developers. Since descriptions of medical symptoms and their diagnosis vary greatly over different countries, as well as that they may vary in their actual details such as additional circumstances to be considered, e.g. for a pregnant woman vs. for a child etc., elegant and efficient workflows need to be defined in order to integrate those heterogeneous data resources. These inconsistencies together with the sensibility and confidentiality of the information [4] shared make this task not only important but in fact a difficult endeavour. The approach chosen within ViroLab based on the development of a middleware system containing a set of virtualization services that hides the distributed and heterogeneous data resources and their internals from the users and guarantees data access in a transparent, consistent and resource-independent way.

To facilitate information exchange among participants and to ease the storage of (bio)medical data sets, particularly in the field of HIV analysis and treatment, the ViroLab team members decided to use and set up a specifically developed HIV database management system the RegaDB HIV Data and Analysis Management Environment [12] developed by the Rega Institute of the Katholieke

Universiteit Leuven either at each data provider site or at different dedicated locations the so-called collaborative (proxy) databases. RegaDB provides a kind of data storage system including some software tools, which may be installed and managed locally, to store clinical data related to HIV treatment. It aims to support clinicians and researchers in their daily work by delivering a free and open source software solution. For researchers the objective is to offer several tools for HIV sequence analysis and to enable and disburden collaborations between researchers of different hospitals and institutions. Clinicians benefit from this system through the visualization of genetic data, relevant drugs, and algorithms in a well arranged form and the automatic generation of drug resistance interpretation reports. Following the described approaches (having a unified data access point) may alleviate integrational difficulties and ensure beneficial properties for both data providers and medical experts.

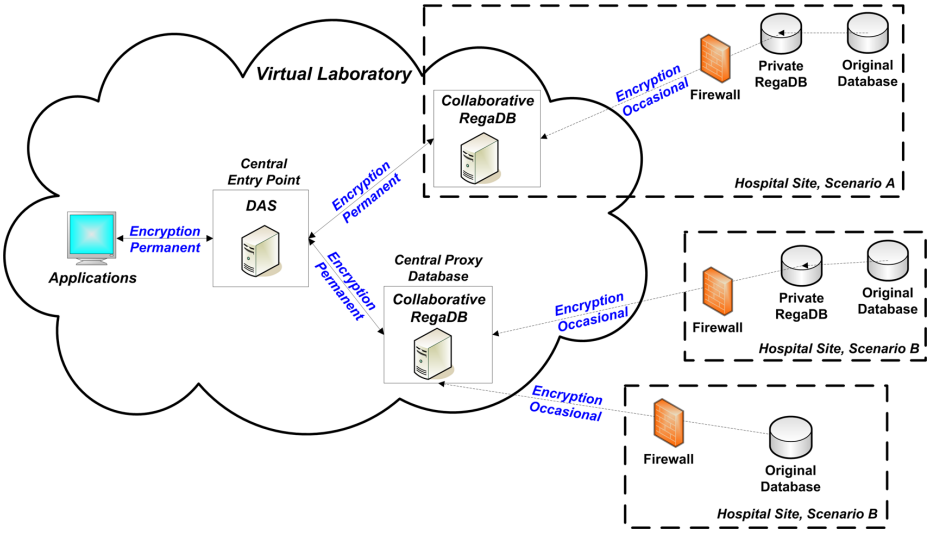

Fig. 1. ViroLab data integration architecture

As depicted in Fig. 1, every data provider can either host a collaborative RegaDB installation within their trusted region, the so-called Demilitarized Zone (DMZ), outside their institutional firewall(s) or upload data onto one of the centrally managed collaborative RegaDB proxies installed at some trusted third parties. Direct external access into the hospitals security regions is not required anymore and the existing database system can still be used independently from sharing any data within the virtual laboratory. Additionally, a hospital can also set up a private RegaDB for its daily purposes and internal data management. To actually contribute data to the virtual environment, the established database schemes need to be converted into the RegaDB schema before data can be stored within any of the RegaDB installations. The migration is done by exporting

data from the original database and converting that data extract through a custom script into the latest schema. This procedure can be conducted repeatedly over time at the discretion of the corresponding database administrator(s) and occurs within each hospital. Data anonymization can also occur during that data conversion or alternatively before transferring data from a private RegaDB onto a collaborative one.

4 Exposing the Aggregated Data Sets

Once the integration of heterogeneous data resources has been realized, the queried data sets need to be somehow aggregated and securely published for making them accessible within different applications and/or systems. As mentioned earlier, a particular set of services is required for dealing with multiple resources simultaneously and for ensuring transparent access to corresponding data storages. This set of services, the so-called Data Access Services (DAS), has been designed and implemented according to the specific needs and requirements for biomedical data and confidential information [4] shared among the laboratory users. Basically, for supporting a wide range of end-user applications and allowing a smooth interaction with several virtual laboratory runtime components, the services capabilities implement standard Web Service interfaces that can be independently used of the underlying infrastructure/technologies and that can be easily extended and modified due to the application developers needs. To provide a certain level of flexibility and reliability, DAS has been separated into stand-alone containers, each serving a specific purpose in the overall services functionality. For exposing and querying single or federated databases, the data handling subsystem is responsible for coordinating all data activities including resource identification, database queries, data consultation and transformation, and finally result message preparation and delivery [3]. Observing data confidentiality and ownership but also guaranteeing a secure message transfer, the security handling module takes existing security standards provided amongst others by the Globus Toolkit [13] or Shibboleth [2], and extends those mechanisms with own highly sophisticated features to meet the requirements for sharing single data sets containing confidential patients information. Principally, additional and stronger authorization mechanisms shall be exploited in order to limit, deny or permit access to different resources in a dynamic and flexible way. Attributes of users including their organization, department, role, etc. are used to administer access rights to resources and services. To explain the complex interplay of the major DAS subsystems (data and security handling) we briefly demonstrate a common use case how one can submit a distributed data query to all available data resources concurrently and how individual results are processed before users finally receive the consolidated output. Typically, doctors want to use the virtual environment for requesting patient information including genetic data such as nucleotide sequences or mutations in order to predict any possible drug resistance for their according case. They simply want to retrieve all relevant information without requiring any specific expertise in computer science. Therefore,

the way to get to the data must be kept simple and transparent for them but should be as self-explaining as possible. The capability provided by DAS for submitting such queries requires a standard SQL query as input and then automatically performs the following actions: checks which resources are available; requests data resource information such as database type, keywords, etc. of each available resource and compares the resources keywords (pre-defined by resource owner like patients, mutations, sequences, etc.) to the table names stated in the query. If corresponding matches are found, each resource is sequentially queried using the given statement. Finally, the results are merged and the resource ID is added to each new data row as an additional primary key to uniquely identify the origin of each single data set.

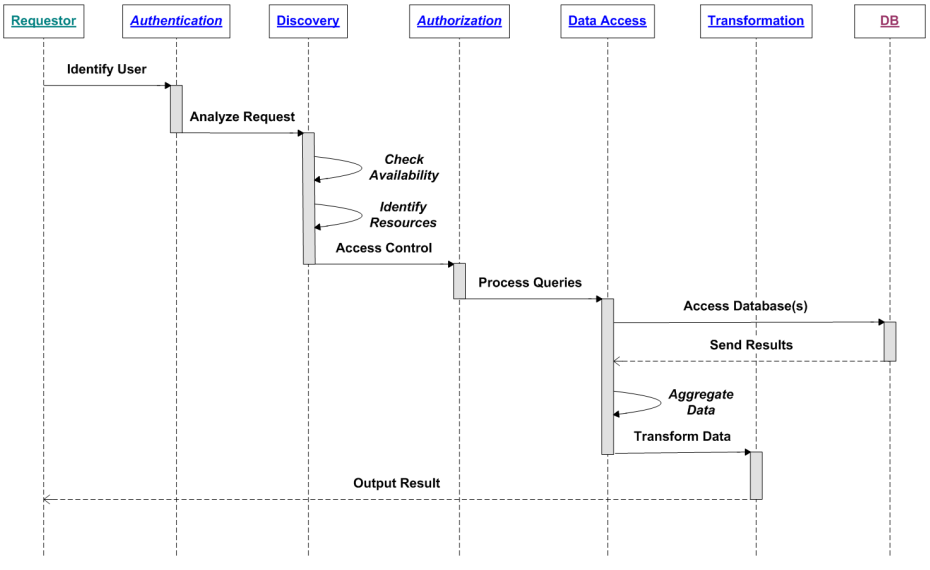

Fig. 2. A typical data access use case within the ViroLab scenario

In Fig. 2, the above-mentioned use case together with the corresponding actions is highlighted again. Each single step - starting with the users request up to the response sent back by the DAS - is depicted within this chain by one specific block. The resource identification, the pure data access, and finally the application-dependent transformation are handled by the data handling module whereas all security operations are carried out by the security handling module in cooperation with the entire security framework deployed within the virtual laboratory. How a doctor can really get to the data and how the DAS is interfaced from other laboratory components in order to interact and in particularly send such queries, is explained in the next section. [11]

5 Interfacing and Manipulating Medical Data in the Virtual Laboratory

The Data Access Services are an integral part of the ViroLab Virtual Laboratory [9], but it cannot be directly interfaced by the ViroLab user layer. In fact, the intricacies associated with the invocation of complex services, manipulating security credentials, submitting queries and processing the received replies would add undue complexity to the VL scripting language, which is designed to be as simple as possible and contain little to no syntax involved with the actual interoperation with the underlying services. Hence, a separate module of the ViroLab runtime system, called the Data Access Client (DAC), is implemented. The DAC has several important functions: carrying out all communications with the Data Access Services, including user authorization, submission of queries and importing results into the context of the ViroLab experiment scripts; presenting data to experiment developers in a way which can be easily manipulated in the experiment host language [18]; interfacing with external "ad hoc" data sources, which are not aggregated under the Data Access Services (for instance, scratchbook databases and P2P storage frameworks), and finally providing a data manipulation layer tool for the submission and storage of ViroLab experiment results.

The Data Access Client is implemented as a JRuby library, which is automatically imported into each experiment script and then, in turn, interfaces with the underlying services such as the Data Access Services. In addition, the interface of the Data Access Client is also exposed directly by the GridSpace Engine (the runtime component of the ViroLab Virtual Library), where it can be utilized by other, external tools which are part of the ViroLab Virtual Laboratory (such as the provenance tracking service).

The basic tenet of the Data Access Client is simplicity of use. Therefore, the API offered by the client to scripting developers is as simple as possible. In order to initiate communication with a data source, all the developer has to do is to instantiate an object of the class DACConnector with the proper arguments. It is only necessary to specify the type of data source and the address (URL) at which the source is located. If there are multiple sources deployed at a given address, it is necessary to select one by providing its schema name.

Fig. 3 presents interaction between the experiment developer and the data access client. Once a data source object is instantiated, the user can use it to import data from the data source, manipulate this data and write data back to the source, if the source supports this functionality. Queries can be formulated in SQL, for standalone data sources and databases aggregated under DAS. Work on integrating XQuery for XML-based data sources is ongoing. The API of the DAC, along with the GSEngine itself, is described in [9]. Thus, the Data Access Client is fully integrated with the JRuby API presented to experiment developers.

As data coming from hospital sources is typically secured against unauthorized access, the Data Access Client must support the authentication methods in use by the Data Access Services (conforming to the overall ViroLab policy on data handling). Since authentication and authorization security in ViroLab

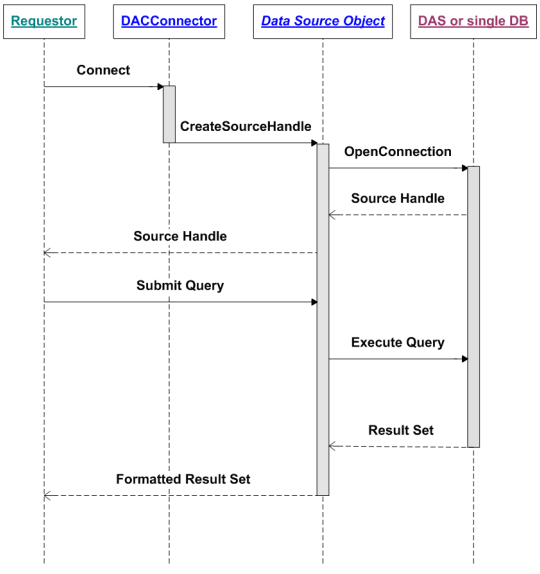

Fig. 3. Interfacing external data resources via DAC

is provided by the Shibboleth attribute-based framework, DAC must authorize itself with DAS prior to retrieval of actual data. In this respect, the DAC relies on the GridSpace Engine (the runtime component of the Virtual Laboratory) to acquire the security handle of the current user, then presents this handle to the Data Access Services so that proper authorization can take place. This process is entirely transparent from the point of view of the experiment user and it does not require the experiment developer to insert additional security-related code in the experiment script.

6 Results

At present, the ViroLab Virtual Laboratory is being applied to a number of applications involving research on the HIV virus. A list of experiments being conducted with the use of the presented solutions can be found at [22]. A representative application in this scope is the "From Genotype to Drug Resistance" framework. This application starts with aggregated data representing viral genotype, then matches this genotype to a given set of rules regarding the susceptibility of the virus to various drugs. In the end, this application is able to determine the most effective course of treatment for a given virus mutation and recommend drugs to be administered to a given patient. In order to ascertain viral susceptibility to various drugs, this application relies on the presented data access subsystem to interface with participating hospitals, securely collect viral genotype data and present it to experiment developers in a uniform manner,

using a common schema. This process is further described in [12] and is now being successfully applied in the ViroLab project [21].

7 Summary and Future Work

The presented data access solutions form an integral part of the ViroLab Virtual Laboratory and enable medical researchers to conduct studies in the field of viral diseases treatment as well as other associated areas of medical science.

Current work on the Data Access Client focuses on extending its functionality to provide a backend for the submission, storage and retrieval of VL experiment results. This requires interfacing with WebDAV data storage repositories, as well as with the ProToS Provenance Tracking System [5] which will be used to store metadata describing such results as susceptibility of the HIV virus to various forms of treatment. Once complete, this extension will provide a layer of persistence to all data generated with the use of the ViroLab Virtual Laboratory. We are also conducting research into a potential uniform data schema for all types of data sources used in the ViroLab Virtual Laboratory.

Future developments planned for the Data Access Services will mainly enhance reliability and scalability of the individual services capabilities as well as increase the data submission performance through processing queries in parallel instead of submitting requests one after another. Finally, facilitating the management of access control policies, the corresponding capabilities of the security handling unit will be integrated with a nice and user-friendly graphical user interface allowing the fast and dynamic generation, change, and upload of access control policies for certain data resources in order to provide more flexibility in administering distributed resources within a collaborative working environment.

Acknowledgements: This work is supported by the EU project ViroLab IST-027446 and the related Polish SPUB-M grant, as well as by the EU IST CoreGrid project.

References

1. Antonioletti, M., Atkinson, M.P., Baxter, R., Borley, A., Chue Hong, N.P., Collins, B., Hardman, N., Hume, A., Knox, A., Jackson, M., Krause, A., Laws, S., Magowan, J., Paton, N.W., Pearson, D., Sugden, T., Watson, P., Westhead, M.: The Design and Implementation of Grid Database Services in OGSA-DAI. Concurrency and Computation: Practice and Experience 17(2-4), 357–376 (2005)
2. Assel, M., Kipp, A.: A Secure Infrastructure for Dynamic Collaborative Working Environments. In: Proceedings of the 2007 International Conference on Grid Computing and Applications (GCA 2007), Las Vegas, USA (June 2007)
3. Assel, M., Krammer, B., Loehden, A.: Data Access and Virtualization within ViroLab. In: Proceedings of the 7th Cracow Grid Workshop 2007, Cracow, Poland (October 2007)
4. Assel, M., Krammer, B., Loehden, A.: Management and Access of Biomedical Data in a Grid Environment. In: Proceedings of the 6th Cracow Grid Workshop 2006, Cracow, Poland, October 2006, pp. 263–270 (2006)

5. Balis, B., Bubak, M., Pelczar, M., Wach, J.: Provenance Tracking and Querying in ViroLab. In: Proceedings of Cracow Grid Workshop 2007, Krakow, Poland (December 2007)
6. Bubak, M., Harezlak, D., Nowakowski, P., Gubala, T., Malawski, M.: Appea: A Platform for Development and Execution of Grid Applications e-Challenges - Expanding the Knowledge Economy. IOS Press, Amsterdam (2007) ISBN 978-1-58603-801-4
7. Cannataro, M., Guzzi, P.H., Mazza, T., Tradigo, G., Veltri, P.: Using ontologies for preprocessing and mining spectra data on the Grid. Future Generation Computer Systems 23(1), 55–60 (2007)
8. Cannataro, M., Talia, D., Tradigo, G., Trunfio, P., Veltri, P.: SIGMCC: A system for sharing meta patient records in a Peer-to-Peer environment. Future Generation Computer Systems 24(3), 222–234 (2008)
9. Ciepiela, E., Kocot, J., Gubala, T., Malawski, M., Kasztelnik, M., Bubak, M.: GridSpace Engine of the Virolab Virtual Laboratory. In: Proceedings of Cracow Grid Workshop 2007, Krakow, Poland (December 2007)
10. Frattolillo, F.: Supporting Data Management on Cluster Grids. Future Generation Computer Systems 24(2), 166–176 (2008)
11. Gubala, T., Balis, B., Malawski, M., Kasztelnik, M., Nowakowski, P., Assel, M., Harezlak, D., Bartynski, T., Kocot, J., Ciepiela, E., Krol, D., Wach, J., Pelczar, M., Funika, W., Bubak, M.: ViroLab Virtual Laboratory. In: Proceedings of Cracow Grid Workshop 2007, Krakow, Poland (December 2007)
12. Libin, P., Deforche, K., Van Laethem, K., Camacho, R., Vandamme, A.-M.: RegaDB: An Open Source. In: Community-Driven HIV Data and Analysis Management Environment Fifth European HIV Drug Resistance Workshop, Cascais, Portugal, March 2007, vol. 2007-2, published in Reviews in Antiretroviral Therapy (2007)
13. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Globus Project (2002), `http://www.globus.org/research/papers/ogsa.pdf`
14. Marovic, B., Jovanovic, Z.: Web-based grid-enabled interaction with 3D medical data. Future Generation Computer Systems 22(4), 385–392 (2006)
15. Zazzi, M., et al.: EuResist: exploration of multiple modeling techniques for prediction of response to treatment. In: Proceedings of the 5th European HIV Drug Resistance Workshop, European AIDS Clinical Society (2007)
16. Enabling Grids for E-sciencE, `http://public.eu-egee.org/`
17. Gridipedia: The European Grid Marketplace, `http://www.gridipedia.eu/`
18. JRuby: A reimplementation of the Ruby language in pure Java, `http://jruby.codehaus.org/`
19. The Open Grid Forum, `http://www.gridforum.org/`
20. The SIMDAT project, `http://www.scai.fraunhofer.de/`
21. The ViroLab Project Website, `http://www.virolab.org/`
22. The ViroLab Virtual Laboratory Web Portal, `http://virolab.cyfronet.pl/`