Design and development of vulnerability management portal for DMZ admins powered by DBPowder

Tadashi Murakami[1,*]

[1]High Energy Accelerator Research Organization (KEK), Japan

Abstract. It is difficult to promote cyber security measures in research institutes, especially in DMZ networks that allow connections from outside network. This difficulty mainly arises from two types of variety. One is the various requirements of servers operated by each research group. The other is the divergent skill level among server administrators. Unified manners rarely fit managing those servers. One of the solutions to overcome the above mentioned difficulties is vulnerability management. To overcome these challenges, There are two possible approaches. One of the options is to offer a simple and powerful vulnerability management service to the administrators of the DMZ hosts (DMZ admins). The other is to facilitate flexibility and efficiency in the development process of the service. To achieve these requirements, we designed and developed a vulnerability management portal site for DMZ admins, named DMZ User's Portal. This paper describes the design of DMZ User's Portal and the development process using a development framework, named DBPowder. Using the DMZ User's Portal, each DMZ admin can perform a vulnerability scan on his/her own servers with ease. In other words, this delegates security vulnerability discovery and responsibility to individual DMZ admins that improve security awareness for them. Then, each DMZ admin can grasp and manage the security by himself/herself. The 13-year result from vulnerability scans show that the status of security in the KEK-DMZ has been kept in good conditions. Also, we are developing DBPowder object-relational mapping (ORM) framework to improve the flexibility and efficiency in the development process of DMZ User's Portal.

1 Introduction

In KEK, there are various research groups within the fields of high-energy physics, material physics, and accelerator physics, and they offer various information and communication technology (ICT) services to various researchers around the world. Each field of physics has its own way to proceed their research.

Their services are also diverse, including experiments, GRID services, public relations, and login shell, among others (Figure 1). Many of these services cannot fit within the standard ones provided by the computing research center. In the KEK-DMZ network, there are over 300 individual hosts, which are managed by about 100 administrators (DMZ admin). Each of the hosts has its own circumstances due to historical reasons. As a result, it is difficult to apply general security standards. Command-hierarchical manner is not suitable for research

*e-mail: tadashi.murakami@kek.jp

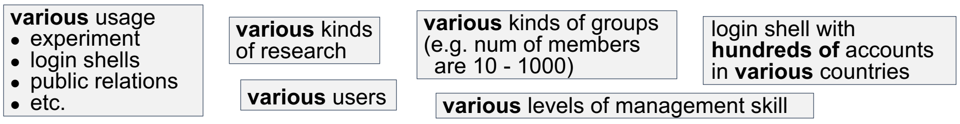

various usage
• experiment
• login shells
• public relations
• etc.
various kinds of research
various users
various kinds of groups (e.g. num of members are 10 - 1000)
various levels of management skill
login shell with **hundreds of** accounts in **various** countries

Figure 1: Various demands for DMZ hosts.

institutes since it is difficult to cover these various circumstances. Instead, each of the DMZ admins has his/her responsibility for security management, and they have to spend a lot of effort to maintain their security.

We have a vulnerability scanner, Tripwire IP360, which is used to analyze hosts and evaluate each one's vulnerabilities in the form of a score. The scanner is proprietary and has rich functions, but it is too intricate for non-experts in security to utilize.

To address these challenges, we developed DMZ User's Portal site, in 2007, that utilizes the vulnerability scanner. Since then, we have operated and improved the portal site. All of the owners of DMZ hosts have their own accounts.

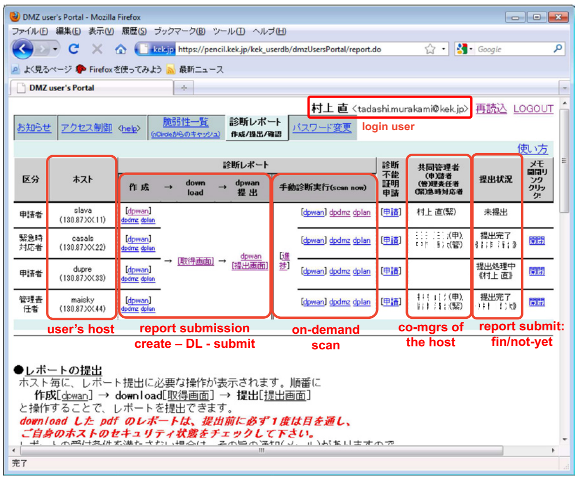
Using DMZ User's Portal, each DMZ admin can perform a vulnerability scan on his/her own servers with ease. Then, each DMZ admin can grasp and manage the security by himself/herself. This paper shows the design and development issues of DMZ User's Portal.

2 Application design of the vulnerability management portal for DMZ admins – DMZ User's Portal

2.1 Overview of DMZ User's Portal

The vulnerability scanner Tripwire IP360 evaluates each host's vulnerabilities in the form of a score. If the score is over 1000, its vulnerability is regarded as severe. The scanner is proprietary and has rich functions, but it is too intricate for DMZ admins to use because of these rich functions themselves.

To tackle with these challenges, we developed the first version of DMZ User's Portal site in 2007 that wraps the functions and promotes security self-management for DMZ admins.


DMZ user's Portal - Mozilla Firefox
ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
kek.jp https://pencil.kek.jp/kek_userdb/dmzUsersPortal/report.do
Google
よく見るページ Firefox を使ってみよう 最新ニュース
DMZ user's Portal
村上 直 <tadashi.murakami@kek.jp>
再読込 LOGOUT
login user
お知らせ
アクセス制御 <help>
脆弱性一覧 (nCircleからのキャッシュ)
診断レポート 作成/提出/確認
パスワード変更
使い方
user's host
report submission create – DL - submit
on-demand scan
co-mgrs of the host
report submit: fin/not-yet
完了

区分	ホスト	診断レポート 作成 → download → dpwan 提出	手動診断実行(scan now)	診断不能証明申請	共同管理者 (申)請者 (管)理責任者 (緊)急時対応者	提出状況	メモ欄開閉リンククリック!
申請者	slava (130.87.XX.11)	[dpwan] dpdmz dplan	[dpwan] dpdmz dplan	[申請]	村上 直(緊)	未提出	
緊急時対応者	casals (130.87.XX.22)	[dpwan] dpdmz dplan	[dpwan] dpdmz dplan	[申請]	[redacted](申), [redacted](管)	提出完了 《[redacted]》	開閉
申請者	dupre (130.87.XX.33)	[dpwan] dpdmz dplan	[dpwan] dpdmz dplan	[申請]		提出処理中 《村上 直》	開閉
管理責任者	maisky (130.87.XX.44)	[dpwan] dpdmz dplan	[dpwan] dpdmz dplan	[申請]	[redacted](申), [redacted](緊)	提出完了 [redacted]	開閉

→ [取得画面] → dpwan [提出画面]
[進捗]

●レポートの提出
ホスト毎に、レポート提出に必要な操作が表示されます。順番に
作成[dpwan] → download[取得画面] → 提出[提出画面]
と操作することで、レポートを提出できます。
download した pdf のレポートは、提出前に必ず１度は目を通し、
ご自身のホストのセキュリティ状態をチェックして下さい。

Figure 2: Main page of DMZ User's Portal.

Since then, we have been improving the portal site with agility. All of the owners of DMZ hosts have their own accounts. The screen of the main page is shown in Figure 2.

There are three main features in DMZ User's Portal. One of the features provides an easy-to-operate user-interface to DMZ admins so that they can manage and handle the vulnerabilities by themselves. Using the portal, DMZ admins can use the vulnerability scanner easily with one click to run a security analysis of the owner's host. The results can be downloaded as a PDF report.

Another feature helps DMZ admins from both viewpoints of support side and command-hierarchy side in harmony. On the support side, DMZ admins can conduct the vulnerability scan on their own, and the portal collects and aggregates the information to maintain the security. On the command-hierarchy side, the portal helps the duty for DMZ admins to self-inspect their hosts annually, by providing a management interface. In other words, this delegates security vulnerability discovery and responsibility to individual DMZ admins that improve security awareness for them. It also brings scalability into security management.

The third feature provides feedback mechanisms of the vulnerability information of their host in multiple and continuous ways. Three scanners are set on the WAN, DMZ, and LAN networks, which assume attacks from each of their locations. While DMZ admins can conduct an on-demand scan, the portal conducts a regular scan of all DMZ hosts once a week. When serious vulnerabilities are found on a DMZ host, the portal notifies the owner of the host of the information through a weekly email. While DMZ admins can browse the vulnerability list all together, they can also download the vulnerability report of each host individually. As shown above, various feedback mechanisms help DMZ admins to determine the priority to measure, and security management can be executed by each user.

2.2 Security self-inspection performed by DMZ User's Portal

We have been performing annual security self-inspection since 2007. In the self-inspection, DMZ admins use DMZ User's Portal themselves to check their hosts and submit reports. The reports are inspected by our security management committee.

In the self-inspection, each DMZ admin performs a vulnerability analysis from the scanner in the WAN using DMZ User's Portal. If the analysis detects any vulnerability that has a score of over 1000 point, the DMZ admin modifies the vulnerability and retries the analysis. The DMZ admin submits a report when there are no vulnerabilities over 1000 point

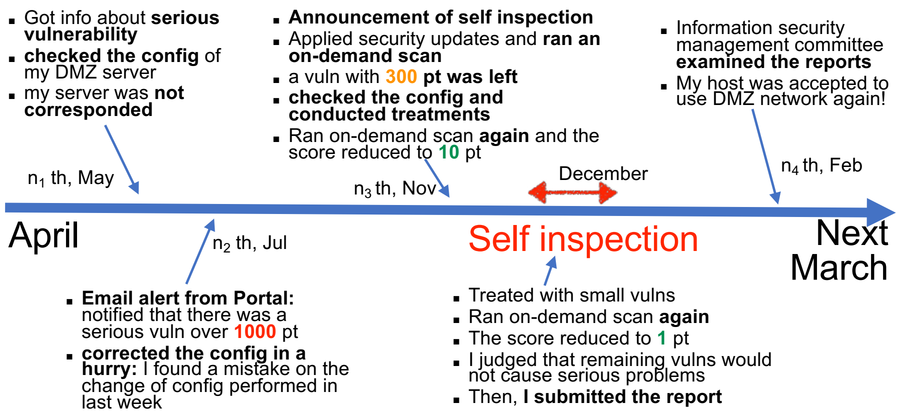

Got info about serious vulnerability
checked the config of my DMZ server
my server was not corresponded
Announcement of self inspection
Applied security updates and ran an on-demand scan
a vuln with 300 pt was left
checked the config and conducted treatments
Ran on-demand scan again and the score reduced to 10 pt
Information security management committee examined the reports
My host was accepted to use DMZ network again!
n1 th, May
n3 th, Nov
December
n4 th, Feb
April
n2 th, Jul
Self inspection
Next March
Email alert from Portal: notified that there was a serious vuln over 1000 pt
corrected the config in a hurry: I found a mistake on the change of config performed in last week
Treated with small vulns
Ran on-demand scan again
The score reduced to 1 pt
I judged that remaining vulns would not cause serious problems
Then, I submitted the report

Figure 3: An example story of admin tasks over one year, from April to March (JFY).

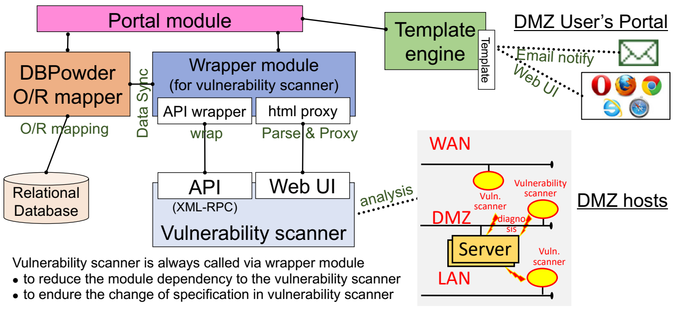


Figure 4: System architecture of DMZ User's Portal.

remaining. If the DMZ admin supposes that the result includes any false-positives, he/she can submit a supplemental report that describes how each of the false-positives is already fixed, with evidence. Besides, the DMZ admin submits Q/A reporting sheets. Finally, their submitted reports and sheets are investigated and examined by the KEK security management committee.

Figure 3 shows an example story of admin tasks over one year, from April to March (JFY). If there is a serious vulnerability that has a score of over 1000 points, DMZ User's Portal sends an alert email weekly to the owner of the host, as shown in 'n_2 th, Jul'. While KEK security team also grasps the weekly alert mail, in these years DMZ admins modified such vulnerabilities by themselves in one or two weeks.

3 System design of DMZ User's Portal

3.1 System architecture

Figure 4 shows the system architecture of DMZ User's Portal. The point is that it has a wrapper architecture. DMZ User's Portal mainly uses four middlewares: vulnerability scanner, relational database, web server, and email sender. All of these are always called via wrapper modules of their own. This reduces the module dependency.

DMZ User's Portal has four wrapper components for the aforementioned middlewares: Portal module, Wrapper module for the vulnerability scanner (Wrapper module in short), DBPowder O/R mapper (DBPowder in short), and Template engine. The Portal module uses the other three core components and organizes the functions of DMZ User's Portal. The Wrapper module receives all requests related to the vulnerability scanner and handles them. DBPowder receives all requests related to the relational database and handles them. DBPowder also helps the development around the relational database. Template engine generates the web UI and email texts.

The vulnerability scanner also manages the data regarding accounts, hosts, results of vulnerability analysis, and so on. While the data can be accessed via XML-RPC or http in the vulnerability scanner, the performance is quite poor. The Portal module uses DBPowder to synchronize the data with the vulnerability scanner and realizes the data access with reasonable performance.

3.2 Wrapper module for the vulnerability scanner – flexibility and maintainability

The vulnerability scanner, Tripwire IP360, has an API with XMLRPC. However, the API does not have some of the essential functions such as the one to download PDF reports. To compensate for this, we developed a wrapper module that has an XMLRPC wrapper and html proxy. At the same time, to reduce the module dependencies on a specific vulnerability scanner, only the wrapper module can access the vulnerability scanner directly using API and https. It also adds a tolerance to the change of specification in the vulnerability scanner.

The flow of the processes of the XMLRPC wrapper is as follows: receive a Java method call, convert it to an XMLRPC call to the vulnerability scanner, receive the XMLRPC return value, convert it to Java class object, and return the object. The flow of the processes of the html proxy is as follows: receive a Java method call, convert it to an http request for the vulnerability scanner and send it, receive the http response that contains http headers and html from the vulnerability scanner, parse and interpret the html to extract the return value, and return the value in the form of Java class object.

In the wrapper module, we also developed several test case modules to check the API wrapper and html proxy. It is particularly useful in the update of the vulnerability scanner. The test case modules find the errors according to the update. When the modification of the errors is finished, the wrapper module is ready to follow the update of the vulnerability scanner.

3.3 Template engine of html and email

When interacting with the portal users, such as those through a web interface and email notification, it is preferable for the contents to be easy to edit. Therefore in DMZ User's Portal, the contents are separated from the codes into templates. At the same time, the templates can be switched using the hostname in which the program of DMZ User's Portal is executed, without changing the configuration in each site.

4 Development of DMZ User's Portal, powered by DBPowder

4.1 DBPowder: object-relational mapping (ORM) framework

In DMZ User's Portal, a database is an important component for managing the data related to accounts, hosts, results of vulnerability analysis, and so on. We introduced DBPowder Object-relational mapping (ORM) framework [1–3] to bridge the database to the Portal module shown in Figure 4. In DMZ User's Portal, we can use the database, database access codes, and the web codes related to database access, via libraries in DBPowder. In our development, DMZ User's Portal improves DBPowder, and DBPowder improves DMZ User's Portal. The theoretical aspects of DBPowder have been already described in previous studies [1–3]. This section describes the software design and usage aspects of DBPowder.

ORM frameworks contribute to the efficient design, development, and maintenance of (a) persistent classes to deal with the persistent data in an object-oriented language, (b) relational schema to manage the persistent data in the RDB, and (c) data conversion between object states and RDB queries/responses [1]. We developed DBPowder ORM framework and utilize it in the development of DMZ User's Portal.

4.2 Development of DMZ User's Portal with DBPowder

4.2.1 DBPowder-mdl: data schema description language

DBPowder-mdl [1] is a language to describe data schema in a simple and flexible manner. Figure 5 is a simple example of DBPowder-mdl. We can design data schema in the left-hand example of *EER-style* such that:

- A `[user]` can `[register]` plural `[host]`s, and a `[host]` is `[register]`ed by plural `[user]`s.

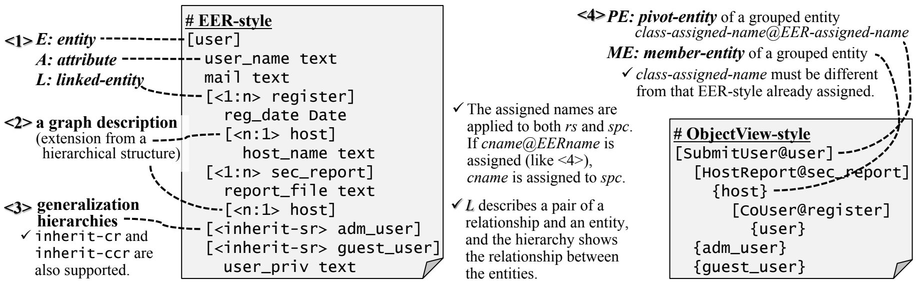

<1> E: entity
A: attribute
L: linked-entity
<2> a graph description (extension from a hierarchical structure)
<3> generalization hierarchies
✓ inherit-cr and inherit-ccr are also supported.
# EER-style
[user]
user_name text
mail text
[<1:n> register]
reg_date Date
[<n:1> host]
host_name text
[<1:n> sec_report]
report_file text
[<n:1> host]
[<inherit-sr> adm_user]
[<inherit-sr> guest_user]
user_priv text
✓ The assigned names are applied to both rs and spc. If cname@EERname is assigned (like <4>), cname is assigned to spc.
✓ L describes a pair of a relationship and an entity, and the hierarchy shows the relationship between the entities.
<4> PE: pivot-entity of a grouped entity
class-assigned-name@EER-assigned-name
ME: member-entity of a grouped entity
✓ class-assigned-name must be different from that EER-style already assigned.
# ObjectView-style
[SubmitUser@user]
[HostReport@sec_report]
{host}
[CoUser@register]
{user}
{adm_user}
{guest_user}

Figure 5: DBPowder-mdl [1].

- A [user] can submit plural [sec_report]s, and a [sec_report] has a [host] in a same context with the [user]-[register]-[host] above.
- The entities of [user], [register], [host], and [sec_report] have their own attributes denoted in Figure 5.
- A [user] can have an inheritance structure, like [adm_user] and [guest_user].

The code generator of DBPowder generates the mapping codes using DBPowder-mdl. By default, the code generator generates classes along with the description of *EER-style*.

It is desirable for data schema to be stable. On the other hand, it is also desirable that the data schema can be used in various way for classes used in an application. We can design classes in the right-hand example (in Figure 5) of *ObjectView-style* such that:

- A [user] has a corresponding class of [User]. Moreover, [user] has another class of [SubmitUser].
- The class [SubmitUser] has a list of [HostReport]s that have a list of [CoUser]s.

The code generator also generates the classes along with the description of *ObjectView-style*.

4.2.2 Helper for database schema modification

In general, database schema modification is not a simple task because the modification does not end with the schema itself. At least, the modification also impacts persistent classes and

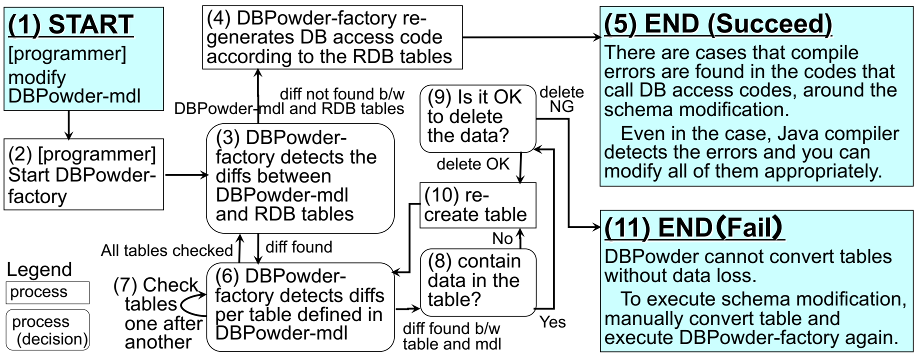

(1) START
[programmer] modify DBPowder-mdl
(2) [programmer] Start DBPowder-factory
(3) DBPowder-factory detects the diffs between DBPowder-mdl and RDB tables
(4) DBPowder-factory re-generates DB access code according to the RDB tables
diff not found b/w DBPowder-mdl and RDB tables
(5) END (Succeed)
There are cases that compile errors are found in the codes that call DB access codes, around the schema modification.
Even in the case, Java compiler detects the errors and you can modify all of them appropriately.
All tables checked
diff found
(6) DBPowder-factory detects diffs per table defined in DBPowder-mdl
(7) Check tables one after another
diff found b/w table and mdl
(8) contain data in the table?
No
Yes
(9) Is it OK to delete the data?
delete OK
delete NG
(10) re-create table
(11) END (Fail)
DBPowder cannot convert tables without data loss.
To execute schema modification, manually convert table and execute DBPowder-factory again.
Legend
process
process (decision)

Figure 6: Flowchart of the helper for database schema modification

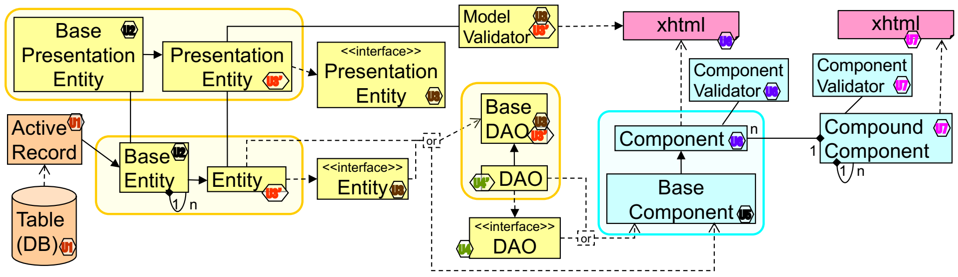


Figure 7: Class structure of DBPowder.

data conversion codes between object states and RDB queries/responses. To make matters worse, many persistent classes are referred to from the application codes. Therefore, many places in the application codes are also impacted by the modification.

DBPowder has the function to assist in schema modification, and it relieves the burden of the task. Figure 6 shows the flowchart of the helper. DBPowder checks the script in DBPowder-mdl, database schema, and classes generated by DBPowder, to keep consistency among them. If a table has one or more data records, any table modification may cause an adverse effect. In such a case, DBPowder tells a developer whether to modify or not.

4.3 Class structure of DBPowder

Figure 7 shows the class structure of DBPowder. Most of the classes are generated per table by DBPowder's code generator according to the description in DBPowder-mdl, while base classes for the classes of presentation entity, entity, and component are common among other classes.

Each of the entities, presentation entities, and components has its own extension point. The extension point is a simple wrapper for developers. Since DBPowder does not modify any extension points, developers can add their own codes without fear of modification by DBPowder.

5 Evaluation

5.1 Statistics of the security self-inspection: submit a report

Figure 8 shows the statistics of report submissions in the security self-inspection in 13 years from 2005 to 2017 (JFY). As the graph indicates, the number of hosts gradually increased

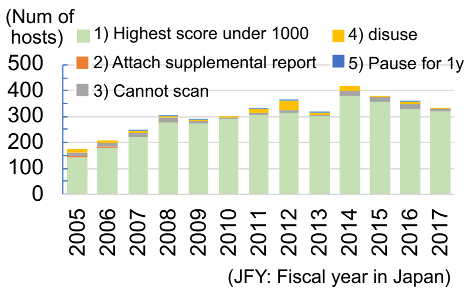


Figure 8: Report submissions in the security self-inspection, from 2005 to 2017 (JFY).

from 2005 to 2014, and the number was inverted in decreasing since 2014. Most of the hosts end in category 1), which means that the submission was completed. In a few cases in 2012 and 2014, there were many hosts in category 4), which means to stop to use the host. In all years, all of the DMZ hosts ended in either of the categories from 1) to 5), indicating that the security self-inspection has been operated successfully.

5.2 Extended DMZ User's portal to other networks

The flexibility shown in Section 3.3 enables us to introduce DMZ User's Portal into the networks which have different operation policy from that of KEK. In 2011, we extended and introduced the portal to the network for J-PARC, a joint project with KEK. In 2017, we introduced the portal to the network for HEPnet-J, another joint project with KEK.

6 Summary

In KEK, various research groups offer various information and communication technology (ICT) services to various researchers around the world. Since each host has its own circumstances and it is difficult to apply general security standards, each host owner has his/her responsibility in security management. Although vulnerability management with a vulnerability scanner is a good solution, it is too intricate for non-experts in security to utilize.

To address these challenges, we developed DMZ User's Portal site that utilizes the vulnerability scanner. There are three main features in DMZ User's Portal. One feature provides a user-friendly interface for DMZ admins to manage and handle the vulnerabilities by themselves. Another feature helps DMZ admins from both viewpoints of support side and command-hierarchy side in harmony. The third feature provides feedback mechanisms of the vulnerability information of their host in multiple and continuous ways. The 13-year result from vulnerability scans show that the status of security in the KEK-DMZ has been kept in good conditions. Also, DBPowder ORM framework brings flexibility and efficiency in the development process of DMZ User's Portal. Flexible implementation has enabled the expansion of DMZ User's Portal to other networks that have different policies.

Acknowledgments

The 13-year history of DMZ User's Portal is based on a lot of fruitful discussions and advices. The author special thanks to: F. Yuasa(KEK), T. Kaneko(KEK), S. Kawabata(KEK), S. Y. Suzuki(KEK), A. Manabe(KEK/J-PARC), A. Tate(J-PARC), Y. Matsuo(J-PARC), H. Ishikawa(J-PARC), information security management committee(KEK), and the users of DMZ User's Portal.

References

[1] T.Murakami, T.Amagasa, H.Kitagawa, DBPowder: A Flexible Object-Relational Mapping Framework Based on a Conceptual Model, *IEEE COMPSAC 2013*, pp. 589 – 598 (2013).

[2] T.Murakami, DBPowder-mdl: Mapping description language between applications and databases, *IEEE/ACIS ICIS 2008*, pp. 127 – 132 (2008).

[3] T.Murakami, DBPowder-mdl: EoD featured and much descriptive domain specific language for O/R mapping, IPSJ Trans. on Databases (TOD), **3**, pp. 46 – 67 (2010) (in Japanese).