

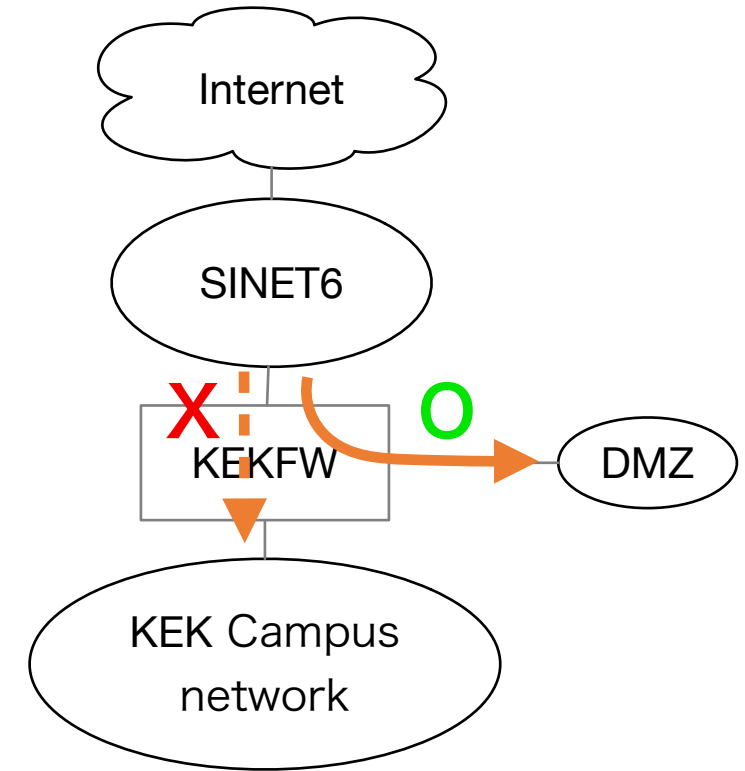
Sustainable Self-Inspection Initiatives for improving Information Security at KEK

R. Yonamine

CRC/KEK

Public servers at KEK

- Public servers at KEK are deployed in DMZ (~300 servers, ~80 admins)
- Those server administrators are supposed to keep their server secure
- **Annual self-inspection campaign for DMZ servers** is one of organizational efforts to increase security
- Submitted reports are to be checked at a dedicated committee



DMZ User's Portal (Past system)

- A system to make a vulnerability scanner (nCircle IP360) easier to handle and submit reports
- Contributed to labor savings for server administrators and efficient compilation work
- The system had been operated and improved for 10 years
- Resulted in significant decrease in the number of vulnerabilities detected by the scanner
 - ➔ **These efforts seem to be working!**



Revising system for self-inspection

- Changes from the time of development
 - Member turnover (The developer had left our team)
 - Vulnerability scanner appliance (nCircleIP360 -> Tenable.sc)
- The Portal became difficult to operate and maintain
- For a few years when the Portal was out of service, we had asked server administrators to fill out a self-inspection form in an Excel file and submit them



Key Roles of DMZ User's Portal

1) Vulnerability scanner Appliance User Interface

➔ Appliance UI itself became more user-friendly

The screenshot displays the Tenable.sc interface for Scan Results. At the top, there is a navigation bar with the Tenable logo and 'Tenable.sc' text. A search bar is present with the text 'Search By CVE'. Below the navigation bar, there are several tabs: Active Scans, Agent Synchronization Jobs, Agent Scans, Scan Results (selected), Attack Surface Domain Discovery, Policies, Audit Files, Credentials, and Freeze Windows. The main content area shows a table of scan results. The table has columns for Name, Scan Policy, Owner, Duration, and Status. A red box highlights the 'Control' buttons (Pause and Stop) and the 'Status' column. The status for the selected scan is 2%.

Name	Scan Policy	Owner	Duration	Status
test_yonamine_ry-mbp2021	#portal.WebComplex#policy	scanner	15m 20s	2%

➔ Wrapper is no longer required

Key Roles of DMZ User's Portal

2) Automatic email notification of detected vulnerabilities

➔ Already migrated to a new notification system

3) Self-inspection system

➔ Self-inspection using Excel files makes analysis and compilation very inefficient

➔ All information to be filled in by server admins themselves

➔ Unavoidable typos on host name or IP make matching against DB difficult

➔ Need a sustainable way

Ideas for New System

- Maintainability is the key
 - Provide 3 functionalities of the Portal in separate systems that are loosely coupled with databases.
 - Web API and popular web frameworks make life easier
 - Node.js, Express as a backend of the web application
 - React as a frontend of the web application
 - Avoid depending on a specific member only



React

System Behaviors

- Forms related to login user appear
- A form per server
- Host name and IP are automatically set
- Co-admins automatically listed
- Filled data is visible to other co-admins
- Forms can be grouped by “tag” names
- Filled data is copiable to other forms
- Jump to a specific form
- Blank check for mandatory questions
- Compilation into PDF or JSON format

The screenshot shows a web interface for managing a host named 'host1' (IP: 192.168.100.1). On the left is a 'Control Panel' with a sidebar containing navigation options like '表示するフォームタグを選択' (Select form tags to display), 'JSONファイル保存' (Save JSON file), and 'セッションタイムアウト' (Session timeout). The main area is the 'Form Display Area', which is horizontally scrollable and contains several sections: 1) 機器情報 (Device information), 2) ソフトウェア (Software), 3) 管理・運用体制 (Management/Operational Policy), and 3-9) その他、補足事項 (Other, supplementary matters). Each section contains various form fields, checkboxes, and text. A table in section 2 lists software in use. A bracket at the bottom of the screenshot groups the sidebar and the main content area.

Control Panel

Form Display Area
(Horizontally scrollable)

Issue on Auto-Notification

- When our security scanners find possible vulnerabilities in servers, the corresponding admins are notified by email, which is implemented in another system.
- Some alarms are false positive
- Some alarms are true positive but there is no way to take measures due to some operational reasons.
- We filter such auto-notification at the request of admins.

Drawback to Filtering Notification

- Once notifications get filtered, they become practically invisible
- Server admins easily forget they have asked to do so while the situation may change over time



- List of filtering comes up in the self-inspection form if any
- Server admins are requested to explicitly their intent to keep the filtering if such a list appears.

host2
[192.168.100.2]

タグ

以下のプラグインによる検出が非通知となっています：

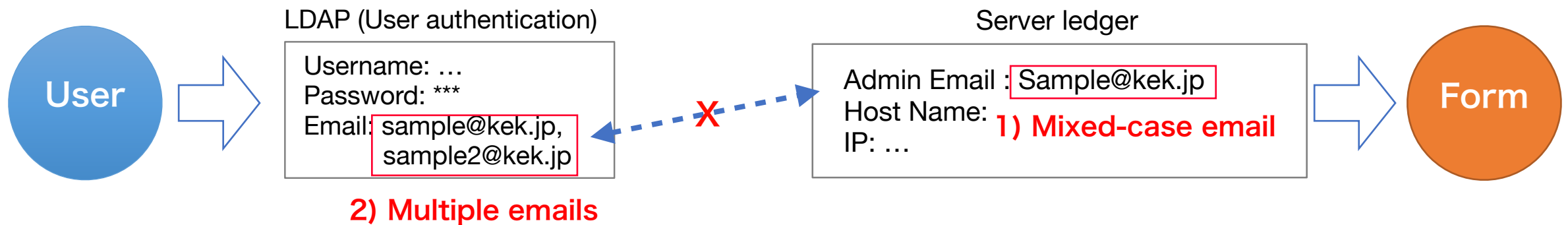
- [42423] CGI汎用SSIインジェクション (HTTPヘッダー) (ポート:443) [wan]
- [42423] CGI汎用SSIインジェクション (HTTPヘッダー) (ポート:80) [wan]

1) 機器情報

機器のメーカー名と製品名。バーチャルマシンのゲストVMの場合は、VMのソフトウェア名、バージョン、および格納先ホストVMのホスト名

Problems discovered during campaign

- The system worked smoothly except for a few specific cases
- Forms not displayed to certain users, because of:
 - 1) Mixed-case registration of email addresses
 - 2) Multiple email addresses registered without priority



- 3) A relatively new method in Javascript : “structuredClone”, which is incompatible with browsers older than ~ 2 years

Remaining issues

- The current situation is the same as that of the past system, which is highly dependent on a single developer.
- Now that we have completed the operational phase and have implemented the minimum necessary functions
- Maintenance and operation are ready to be outsourced

A Trial for Automatic Classification

- The self-inspection requires server admins to list up softwares installed, with classifying into several categories.
- “Others” category often used even when it is not optimal.
- Automatic classification can be easily implemented with Chat-GPT API, and it works nicely than I expected it would!

An example prompt to get GPT’s answers in a format:

- ① “Answer in one-word together with the reason.
- ② Add '@' between answer and reason.
- ③ Classify `${input}` into the following software types: OS, SSH-server, mail-server, ...
- ④ Answer as 'Others' only when none of the above options is appropriate.
- ⑤ Take earlier option if multiple options are possible.”

Summary

- An initiative to improve efficiency and operational stability of self-inspection system for KEK public servers is presented
- Past system for self-inspection campaign highly depend on
 - a specific scanner appliance
 - a single system developer
- New system successfully got through the campaign last year
 - Some improvements on UX
 - But, still highly dependent on a single system developer
- Plan: Outsource maintenance and management of the system