Design and Implementation of Access Control Method Based on Correlation Among Files

Yuki Kodaka
Department of Informatics,
The Graduate University
for Advanced Studies
Tokyo, Japan
email: y_kodaka@nii.ac.jp

Hirokazu Hasegawa
Center for Strategic Cyber
Resilience Research & Development,
National Institute of Informatics
Tokyo, Japan
email: hasegawa@nii.ac.jp

Hiroki Takakura
Center for Strategic Cyber
Resilience Research & Development,
National Institute of Informatics
Tokyo, Japan
email: takakura@nii.ac.jp

***Abstract*—File access control is an effective method for protecting information from unauthorized access both inside and outside an organization. However, conventional methods based on organizational structure have some limitations. Modern business requires flexible access control that reflects the dynamic changes in workflow. Still, it is difficult to achieve the requirement at the same time the prevention of information leakage and destruction due to cyberattacks. Therefore, this paper proposes an access control method based on the correlation among files. The correlation is inferred from users' access behavior within the same group, and access privilege is determined based on the strength of the correlation. This method adapts to changing access needs and prevents unauthorized access by automatically denying access with low file-to-file correlation in a series of accesses. After implementation and verification experiments, it was found that the first determination is the bottleneck of the efficiency of the proposed system. To ensure the feasibility of the proposed system, future work should address this issue.**

***Index Terms*—File access control; Graph theory; Bell-LaPadula model.**

I. Introduction

File access control has long been used as an effective method of protecting an organization's information assets. It prevents unauthorized accesses by users and minimizes information leakage due to cyber attacks. Various access control methods have been proposed and developed [1]–[3].

However, many of the current methods and operations are not flexible enough. Due to changes in the situations, access control loses accuracy over time [4]. In some cases, policymakers (high-level policy architects) and implementers of policy designed by others are separated. And policies are often managed by several persons rather than a single person [5]. These also make flexible operation difficult.

Strict access control is required, especially in environments where sensitive information is handled. For example, the Bell-LaPadula model [6] was proposed to prevent the leakage of information known only to the supervisor to subordinates. However, in many cases supervisors can write to files that their subordinates can read and write to.

According to Proofpoint report [7], the cost of insider threats has surged from $8.30 million in 2018 to $15.38 million in 2022, an 85% increase. In order to mitigate insider threats, not only technical approaches like access control system, but also non-technical approaches like user behavior analytics are needed [8].

To address these issues, we point out two challenges. One is who and how to determine the need for access. In the proposed method, the determination criteria are based on the user's access behavior. Two is how to assign the necessary access privileges for users. Obviously, the assignment of access privileges should be done with caution. Excessive access privileges increase the risk of leakage or destruction. On the other hand, insufficient access privilege affects the ability of users to perform their operations. As a result, it may undermine the efficiency and productivity of the organization.

To solve these problems, we have proposed an access control method based on the correlation among files [13]. The correlation is inferred from user's access behavior. The method automatically determines whether access is allowed or denied based on the degree of the correlation. It responds to access needs based on changing situations. The system automatically denies accesses with low correlation. It prevents excessive expansion of the access privilege. It is assumed that access by malware is an uncorrelated access. Or, even access by an insider is assumed to be uncorrelated if it is not related to the person's business. These accesses are different from legitimate users. This method can prevent such file accesses. In this paper, we have modified the architecture of our system and performed a brief implementation and verification experiment.

This paper is organized in the following sections. Section II refers to related work to this paper. Section III describes the assumptions of the proposed system and issues in file sharing. After that, we explain the design of the proposed system. Section IV describes the implementation and verification of the proposed system. Section V concludes this paper and presents future work.

II. Related work

Users are sometimes denied access to files they need, and administrators are required to modify the access control of the files. They might make a misconfiguration at the modification that gives more access privileges than necessary. Xu et al. investigate how and why such problems occur [10]. Although several reasons for misconfiguration are shown, administrators must solve such problems by themselves, and the possibility of misconfiguration and the burden on administrators remains.

Beckerle and Martucci propose the metrics to evaluate and quantify access control rule sets in terms of security and usability [11]. The metrics helps users generate better rule sets. One of the evaluation indicators is the difference between the owner's intention and the rule set. However, the actual method of getting the intention is out of the scope of the paper.

Mazurek et al. propose reactive policy creation in response to user's access request [12]. The experiment involves sharing files on digital devices at home with people, including supervisors and co-workers. If a user tries to access a resource but lacks sufficient privilege, they can use the proposed system to send a request to the resource owner, who can opt to update their policy and allow the access. This method requires the file owner to make determinations for all unauthorized access.

Shalev et al. propose an improved method for containers that allows monitoring and logging of operations by the system administrator [13]. The operations used by system administrators include not only support by internal IT department employees, but also by third parties such as storage service providers and automated management tools used by the IT department. The system administrator is expected to operate based on user requests (tickets in this paper), but there is no mention of whether or not those requests are required.

Desmedt and Shaghaghi propose an access control method that considers three dimensions: subject, object, and operation, rather than the conventional two dimensions of subject and object [14]. These mainly counter internal threats and provide granular access control by controlling operations. It shows how to implement granular access control, but does not mention how to update access privileges once they have been set.

III. Proposed System

In this paper, we proposed a file correlation-based access control method which is modified from our previous works.

A. Assumption

The proposed method assumes an organization consisting of a hierarchical structure as shown in Fig. 1. This paper calls the largest segment of an organization, such as a department in a typical enterprise, a group. Divided units within the group are called subgroups, and further divided units within a subgroup are called subsubgroups. In the example shown in Fig. 1, each department is a group, and each section is a subgroup.

Fig. 2 shows the assumed access control environment. Generally, an Access Control List (ACL) is implemented with coarse-grained, such as per folder, for groups or subgroups. The access privilege under a folder is determined using the information in a user management database, such as Active Directory (AD). If fine-grained access control is to be implemented, it is set by the file owner or the administrator, but their load becomes significant.

In this paper, resources shared within each group are targeted, and resources shared across groups are out of scope.

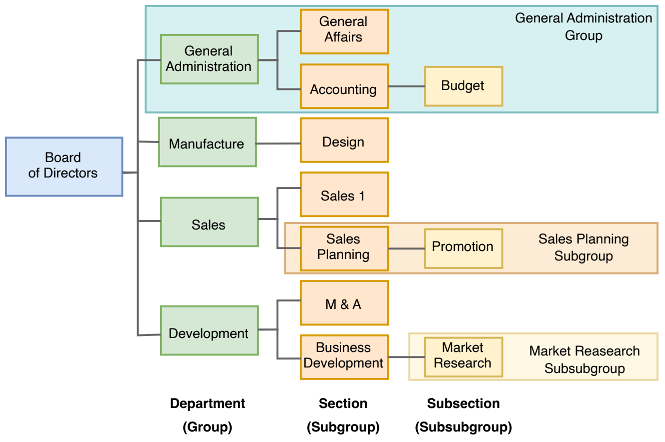

Fig. 1: Example of Organizational Structure

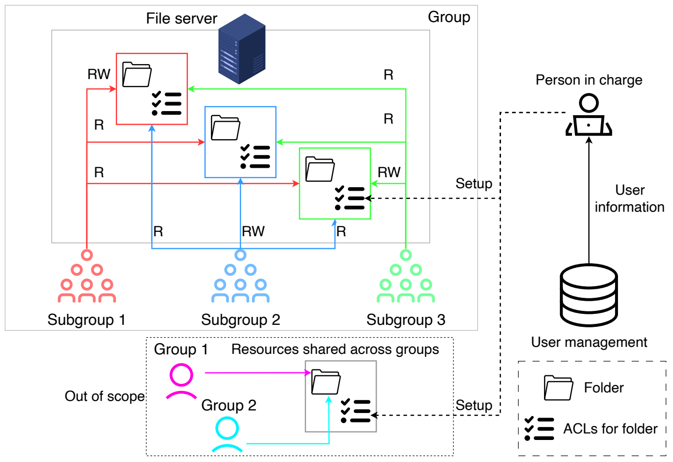

Fig. 2: Assumed Access Control Enviroment

B. Issues in File Sharing

As described in Section I, access control for file sharing involves a risk/benefit tradeoff. Therefore, it is important to balance risks and benefits.

In addition, group or subgroup-based access control alone cannot consider the hierarchical relationship of users in the organization. There is a risk of information leakage through human interaction. It is possible for supervisors to write information in a file that can be read by their subordinates. This could potentially lead to the leakage of information that is known only to the user's rank.

C. Overview of Proposed System

For addressing the file sharing issues, our proposal automatically changes access privileges based on access history for certain period to allow or deny per file, not per folder. It also controls read and write privileges more granularly based on the user's rank. It prevents upper-ranked users from writing confidential information in files that lower-ranked users can read. The proposed method provides hierarchical access control according to the ranks within the organization.

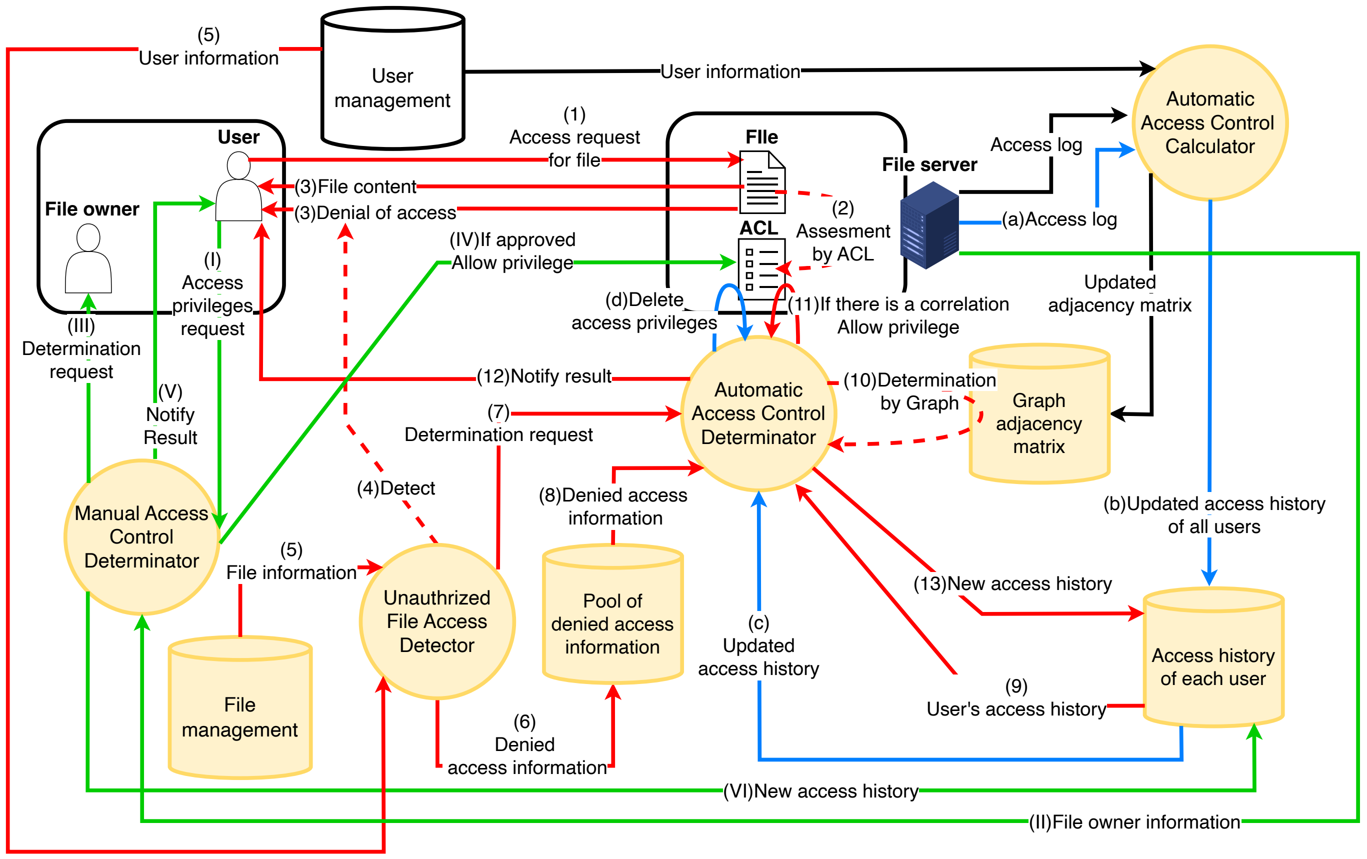
(5) User information
User management
User information
Automatic Access Control Calculator
User
File owner
FIle
(1) Access request for file
(3)File content
(3)Denial of access
File server
Access log
(a)Access log
(2) Assesment by ACL
ACL
(IV)If approved Allow privilege
(I) Access privileges request
(III) Determination request
(d)Delete access privileges
(11)If there is a correlation Allow privilege
Updated adjacency matrix
(12)Notify result
(10)Determination by Graph
Graph adjacency matrix
(V) Notify Result
(7) Determination request
Automatic Access Control Determinator
(4)Detect
(8)Denied access information
(b)Updated access history of all users
Manual Access Control Determinator
(5) File information
Unauthrized File Access Detector
Pool of denied access information
(13)New access history
(c) Updated access history
Access history of each user
File management
(6) Denied access information
(9) User's access history
(VI)New access history
(II)File owner information

Fig. 3: Architecture of Proposed System (yellow colored)

1) Deletion of unnecessary access privileges: The proposed method records the file access history of each user for a certain period in the past (in this paper, one month). If a user has not accessed a file for this period, the access privilege is considered no longer needed, and it is deleted.

2) Addition of necessary access privileges: When a user tries to access a file without access privileges, such access is denied first. Then the proposed method performs an automatic access determination on the denied access. If the results of the determination show that there is a correlation between the files to which the user has access privilege and the denied files, the ACL is changed to "allowed" to access the file. Even if the access is denied as a result of the automatic access determination, the user can request a manual access control determination if the access is truly necessary. In this paper, it is assumed that manual determination is performed by the file owner.

D. Architecture of Proposed System

Fig. 3 shows the architecture of the proposed system. In Fig. 3, the proposed system is colored yellow. It includes the assumed flow of access determination and the source of information necessary for the determination. The proposed system consists of Automatic Access Control Calculator (AACC), Unauthorized File Access Detector (UFAD), Automatic Access Control Determinator (AACD), Manual Access Control Determinator (MACD), and four databases store the target file information, the denied access information, the access history of each user for a certain period, and the adjacency matrix of the graph. Details will be given later, but the overview of the proposed system process is as follows.

- Deletion of unnecessary access privileges (blue line)
  (a) AACC receives access logs for a certain period in the past
  (b) AACC updates access histories of each user
  (c) AACD receives updated access histories and identifies files that the user has not accessed for a certain period
  (d) AACD deletes the user's access privileges from ACL for the identified files
- Automatic determination of access privileges (red line)
  (1) Users access files
  (2) File server determines whether the access is allowed or denied based on ACL set for each file
  (3) If allowed, the user gets the file content
  If denied, the user is notified of denial
  (4) UFAD detects the denied log
  (5) UFAD fetches the target file information from the database for file management and the target user information from the database for user management

(6) If the content of log matches the target files and the users, UFAD store it in the database that stores the denied access information
(7) UFAD requests an access determination to AACD
(8) AACD requests the access information from the database that stores the denied access information
(9) AACD requests the user's access history from the database that stores the access history of each user for a certain period
(10) AACD performs the access determination using graphs
(11) If there is a correlation, AACD allow access privilege to user
(12) AACD notifies the result to the user
(13) AACD adds the newly allowed file to the user's access history

- Manual determination of access privileges (green line)

(I) MACD receives determination requests from users
(II) MACD requests the file owner information from File server
(III) MACD requests the file owner to determine whether the access is allowed or not
(IV) If approved, MACD allow the access privilege to the user
(V) MACD notifies the result to the user
(VI) MACD adds the newly allowed file to the user's access history

The functions of AACC and AACD are described below.

E. Automatic Access Control Calculator (AACC)

Calculator creates graphs utilizing graph theory for correlation determination. The graph infers the correlation among files based on the user's access behavior.

The calculation procedure is as follows. Data is access logs for a certain period, which is the past month (the past 30 days) in this paper.

a) Extract specific information from access logs: Specific information in the access log is recorded as the access log used in the calculation. The specific information is "Timestamp", "AccessType"(Read or Write), "UserName", "Filename". The extracted access logs are sorted by username and time.

b) Categorize access logs by user rank and access type: Extracted access logs are categorized by user rank and access type. Ranks are assumed to be hierarchical. For example, from the top, director, manager, section chief, member. For each user rank, two access logs are categorized. One is the access log of the "Read" access type for users below the same rank. The other is the access log of the "Write" access type for users in the same rank.

c) Create graphs from access history: An example graph is shown in Fig. 4. The graph consists of nodes (V_1, V_2, V_3) and links between nodes (L_{1-2}, L_{2-3}, L_{1-3}). In the graph, nodes represent files. Links represent the correlations among files. The graph is assumed to be undirected. The order of accesses, A-B and B-A are counted as the same.

The graph is calculated using an adjacency matrix. Adjacency means that node i and node j are adjacent to link $i - j$ in the graph. An adjacency matrix is a square matrix used to represent a finite graph. The elements of this matrix indicate if a pair of nodes is adjacent or not in the graph. If so, it indicates the weights of the links between adjacent nodes. An example of an adjacency matrix is shown in Table I.

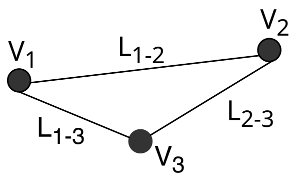

Fig. 4: Example of Graph

TABLE I: EXAMPLE OF ADJACENCY MATRIX

$$\begin{array}{l|cccc} & \text{FileA} & \text{FileB} & \text{FileC} & \text{FileD} \\ \hline \text{FileA} & 0 & 3 & 0 & 1 \\ \text{FileB} & 3 & 0 & 1 & 5 \\ \text{FileC} & 0 & 1 & 0 & 1 \\ \text{FileD} & 1 & 5 & 1 & 0 \end{array}$$

The following procedure is used to calculate link weights.

a. Get a list of files by rank and access type from categorized logs
b. Determine the size of the adjacency matrix from the list
c. Create adjacency matrix initialized to 0
d. Calculate weights of links from access logs
Add link weights between consecutive files in the categorized access history, if the same user accesses different files within a certain period of time (one hour in this case). Furthermore, we add the time inclination shown in (1) based on the timestamp:

$$1 - \left(\frac{D}{D_{\text{max}}}\right)^n \tag{1}$$

where D is the number of days elapsed from the most recent day, D_{max} is the number of calculation days, and n is an adjustment parameter.
e. Normalize weights of links
Let A be the adjacency matrix before normalization, and $S(n)$ be the total weight of the links connected to each node n. Normalization is performed as shown in (2):

$$B(i,j) = \frac{A(i,j)}{S(i)} + \frac{A(j,i)}{S(j)} \tag{2}$$

where $B(i,j)$ is the element at the ith row and jth column of the adjacency matrix after normalization, $A(i,j)$ is the element at the ith row and jth column of the adjacency matrix before normalization, $S(i)$ is the total weight of the links connected to node i, $A(j,i)$ is the element at the jth row and ith column of the adjacency matrix before normalization, and $S(j)$ is the total weight of the links connected to node j. Also, round off to the second decimal place.

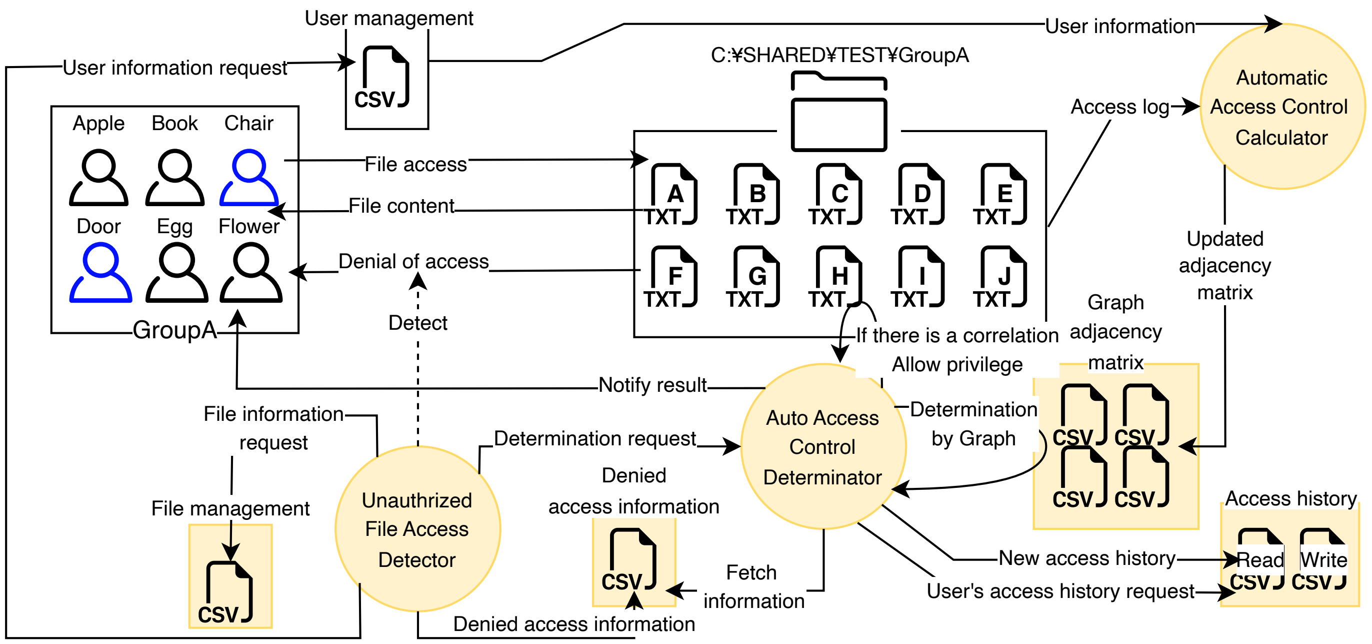

Fig. 5: Implementation of Proposed System (yellow colored)

Table II shows the results of the above calculation steps using Table I as an example.

TABLE II: EXAMPLE OF NOMALIZED ADJACENCY MATRIX

$$\begin{array}{c|cccc} & \text{FileA} & \text{FileB} & \text{FileC} & \text{FileD} \\ \hline \text{FileA} & 0.00 & 1.08 & 0.00 & 0.39 \\ \text{FileB} & 1.08 & 0.00 & 0.61 & 1.27 \\ \text{FileC} & 0.00 & 0.61 & 0.00 & 0.64 \\ \text{FileD} & 0.39 & 1.27 & 0.64 & 0.00 \end{array}$$

F. Automatic Access Control Determinator (AACD)

The determination method is as follows. If the elements of the adjacency matrix exceed a certain threshold, it is considered correlated. The formula is shown in (3). Here, as an example, the threshold is set at 0.8 or higher.

$$\text{Matrix}(\text{File}_{\text{old}}, \text{File}_{\text{new}}) \geq 0.8 \quad (3)$$

where $\text{Matrix}(\text{File}_{\text{old}}, \text{File}_{\text{new}})$ is the correlation between files, File_{new} is the new file to be accessed by user u, and File_{old} is the file already accessed by user u.

IV. Implementation and Evaluation Experiment of Proposed System

A. Implementation of Proposed System

The proposed system was implemented in a brief experimental environment. As the hardware, we used Intel® NUC 8 Pro Kit (NUC8v7PNH). As OS, we used Windows 11 pro edition. The implemented environment is shown in Fig. 5. We set up 6 users, Apple, Book, Chair, Door, Egg, and Flower as general users. We set up 3 users, AACC, UFAD, and AACD, to take on the roles of the proposed system. We created 10 files named A, B, C, D, E, F, G, H, I, J in a folder with the path “C:¥SHARED¥TEST¥GroupA”.

The access log in Windows is “Security” in “Windows Log” (hereinafter referred to as Windows Security Log).

The proposed method needs to notify the determination results to the user. There are several possible ways to notify such as e-mail, text message, phone call or other means. We made notification possible by running the program on both the receiver and the sender of the message during experiment.

The following is a description of the setting for the system and the implementation of each element of the system.

1) Setting File Privileges for Users: The privileges allowed to each user for each file are shown in Table III.

2) Automatic Access Control Calculator (AACC): The graph calculation was set up in two stages. The first stage is calculated at 1:00 a.m. daily using data from 30 days prior to the previous day. The second stage is calculated every hour during business hours, using data up to the present time of the day. After the second stage of calculation, the graphs from the first and second stages were combined and normalized. This is because graph calculation takes a lot of time.

The calculation used Event ID 4663 (An attempt was made to access an object.) from Windows Security Log.

3) Unauthorized File Access Detector (UFAD): Using ”Task Scheduler”, UFAD was triggered by a log that access to a file by the user was denied. The log was Event ID 4656(A handle to an object was requested.) from Windows Security Log. Only failures were collected.

As well as way of notification to the user, UFAD and AACD were contacted by executing the program.

TABLE III: FILE ACCESS PRIVILEGES FOR EACH FILE ALLOWED TO EACH USER

User	FileA	FileB	FileC	FileD	FileE	FileF	FileG	FileH	FileI	FileJ
A	R/W	R/W		R/W	R/W	R	R/W	R/W		R/W
B		R	R/W	R/W	R/W	R/W	R/W	R		R/W
C	R/W	R/W		R/W			R/W	R/W	R/W	R/W
D	R/W	R/W	R	R/W	R/W	R/W			R/W	R/W
E	R/W				R/W	R	R/W	R/W	R/W	R/W
F	R/W	R/W		R/W	R/W	R/W	R/W		R/W	R

UFAD recorded the output access log. In this evaluation, the last 10 access logs were recorded. It also fetched the latest access logs before outputting the denied log. In this case, the latest 10 access logs were fetched. If there is no output in spite of the target log, the logs will be output along with the denied log.

4) Automatic Access Control Determinator (AACD): AACD was listening with a request from UFAD. When it is received, AACD execute processes. The determination was repeated as long as there was information in Pool of denied access information. When there was no more information, AACD waited again in the listening state.

AACD recorded the history of past determinations. In this paper, it was recorded for the past 10 minutes. If the same file was accessed and rejected within 10 minutes, the determination was made only once, and the rest of the accesses were skipped without determination.

5) CSV File for User management: This file recorded information in the following three columns.

- UserName (Apple, Book, Chair, Door, Egg, Flower)
- Rank (Chair and Door is rank 2, the rest is rank 1)
- Group (all users are Group A)

6) CSV File that stores target file information: This file stored a list of target files (A, B, C, D, E, F, G, H, I, J).

7) CSV File for Pool of denied access information: This file stored the denied log information Timestamp, AccessType, UserName, FileName.

8) CSV Files for Graph adjacency matrix: These files stored the adjacency matrices of the graph calculated by AACC.

9) CSV Files for Access history of each user: These files stored the access history of each user for each access type.

B. Evaluation Experiment of Proposed System

In this section, the efficiency of the proposed system is verified as an evaluation experiment. As a measure, the response time was tested. Methodology are shown below.

1) Methodology: Two types of experiments were conducted. First, one to six users simultaneously accessed an unauthorized file. We verified the change in processing time due to the change in the number of users. Second, one to six users accessed unauthorized files at regular intervals from the previous user. In this experiment, the next user accessed the file at 5-second delays. We verified the change in processing time when several users accessed at regular intervals. The number of users was increased from one to six, and each was performed five times.

Access is done by executing a script at a specific time using the task scheduler. The script indicates the name of the file to be accessed. It is a file to which each user does not have privileges. The script for user Apple describes file C. Similarly, the script for Book describes file A, the one for Chair describes file C, the one for D describes file G, the one for Egg describes file B, and the one for F describes file C. The time from when the script is executed to when the determination results are notified to the user is measured as the response time.

TABLE IV shows the data sets used in the graph calculations for the determination. Since the data set up to the previous day has 404 rows, only a summary of the data set is shown. Up to the day is without parentheses and the day is surrounded by parentheses.

2) Results: Experimental results for simultaneous access and access with 5-second delays are shown in Table V. Number of user columns indicates the number of accessing users and the initial letter of the accessing user. For example, 1(A) indicates that a single user named Apple accessed an unauthorized file. The columns 1st trial through 5th trial show the response times for each number of users. If the number of users is 2 or more, the latest response time among users is noted. 5-trial average column shows the average of five trials for the same number of users. Average per user is calculated by dividing the 5-trial average by the number of users in the corresponding row. In this paper, a case is defined as the number of times a determination is made to allow or deny access.

C. Discussion

Table V shows that less determination time is required after the second case. Comparing the simultaneous access of 1(A) and 2(A/B), for example, the average response times per user become 2.8 seconds shorter. This experiment shows that average response time per user decreases as the number of users increases. This indicates the efficiency of the system improves after the second case. However, this also shows the first determination is the bottleneck of the efficiency of the proposed system.

Table V shows that it takes approximately the same amount of time to make a determination from the first case to the sixth case. From this result, it can be inferred that one cycle of determination is completed after the 5-second delays. However, the first case took a little bit longer to determine as same as the result of simultaneous access.

Limitation: In this paper, the proposed system has only been able to verify the situation with six users. However,

TABLE IV: DATA SET SUMMARY FOR GRAPH CALCULATION

User	FileA		FileB		FileC		FileD		FileE		FileF		FileG		FileH		FileI		FileJ	
	R	W	R	W	R	W	R	W	R	W	R	W	R	W	R	W	R	W	R	W
A	7(2)	5(1)	2(1)	0(0)	0(0)	0(0)	3(2)	1(0)	3(1)	0(1)	4(1)	0(1)	4(1)	4(1)	6(0)	4(0)	0(0)	0(0)	0(0)	0(0)
B	0(0)	0(0)	6(0)	0(0)	6(2)	5(1)	6(2)	6(2)	7(1)	6(1)	9(0)	8(0)	7(0)	7(0)	10(1)	0(0)	4(0)	0(0)	0(0)	0(0)
C	5(1)	5(0)	3(1)	1(1)	0(0)	0(0)	8(1)	8(1)	0(0)	0(0)	0(0)	0(0)	2(1)	1(0)	5(1)	5(1)	4(1)	3(1)	0(0)	0(0)
D	2(1)	1(1)	7(0)	7(0)	6(2)	0(0)	10(2)	7(0)	5(2)	4(1)	10(1)	7(0)	0(0)	0(0)	0(0)	0(0)	6(0)	6(0)	0(0)	0(0)
E	9(2)	6(1)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	7(2)	7(1)	6(0)	0(0)	14(1)	13(1)	4(0)	3(0)	9(2)	7(0)	0(0)	0(0)
F	6(1)	3(1)	5(1)	5(1)	0(0)	0(0)	5(1)	4(1)	6(1)	4(0)	5(0)	4(2)	5(2)	4(0)	0(0)	0(0)	3(1)	2(0)	0(0)	0(0)

TABLE V: EXPERIMENTAL RESULTS: RESPONSE TIME

	Response Time (s)						
Number of User	1st trial	2nd trial	3rd trial	4th trial	5th trial	5-Trial Average	Average per User
Simultaneous Access							
1(A)	6.848931	6.220957	6.619395	5.960975	6.688213	6.4676942	6.4676942
2(A/B)	7.961715	7.919432	6.505913	6.712306	7.820161	7.3839054	3.6919527
3(A/B/C)	7.905125	9.562813	7.342283	9.489325	8.582019	8.576313	2.858771
4(A/B/C/D)	8.611524	8.262025	8.348701	8.788181	8.162078	8.4345018	2.10862545
5(A/B/C/D/E)	8.791258	8.952904	9.555604	9.10041	9.461176	9.1722704	1.83445408
6(A/B/C/D/E/F)	11.200058	12.773345	9.905579	12.539686	11.843131	11.6523598	1.942059967
Access with 5s Delay							
1(A)	6.833407	6.63882	6.87596	8.329752	6.656232	7.0668342	7.0668342
2(A/B)	11.263513	11.485306	11.795818	12.237531	11.947039	11.7458414	5.8729207
3(A/B/C)	16.516348	16.310971	15.935014	16.120887	16.474012	16.2714464	5.423815467
4(A/B/C/D)	23.57539	21.183435	22.799851	23.252516	21.822607	22.5267598	5.63168995
5(A/B/C/D/E)	27.278206	27.148512	26.288184	26.199733	27.018846	26.7866962	5.35733924
6(A/B/C/D/E/F)	31.92014	31.139518	31.577622	31.936971	31.464357	31.6077216	5.2679536

it cannot guarantee scalability beyond that. For example, when the number of users reaches 10 or 50 or more, There is a possibility of leaks or duplicates in UFAD detection and processing. Considering the bottleneck of the first case determination, it is necessary to guarantee the scalability and efficiency of UFAD in the future.

In this proposal, the determination is made based on the past access history. It does not solve the problem of how to set the initial settings for newly created files or when a user's department changes. These issues need to be addressed as well.

V. Conclusion and Future Work

In this paper, we propose an access control method that responds to changing situations. It automatically blocks continuous access to files with low correlation. It enables automatic determination and reduces administrative burdens. If there is no or low correlation, detailed manual determination prevents unauthorized access.

After implementation and verification experiments, It was found that the first determination is the bottleneck of the efficiency of the proposed system. In addition, the scalability of it has not yet been verified. To make the system feasible, future work should address the issue of efficiency and scalability. It is also essential to have an adequate data set for verification.

There are some thresholds that need to be set. For example, the extraction period of access logs is set to the past month. The threshold for correlation is a matrix element greater than or equal to 0.8. These thresholds are tentative. They are subject to change depending on the target organization and cyber attack stages. Detailed discussion is required to set them.

Acknowledgment

This work was partially supported by JSPS KAKENHI Grant Number JP19K20268.

References

[1] P. Samarati and S. C. Vimercati, "Access control: policies, models, and mechanisms," Foundations of Security Analysis and Design, R. Focardi, R. Gorrieri, ed., Springer, pp.137-196, 2001.

[2] D. F. Ferraiolo and D. R. Kuhn, "Role-based access control," 15th National Computer Security Conference, pp.554-563, 1992.

[3] V.C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to Attribute Based Access Control (ABAC) Definition and Considerations," U.S. Department of Commerce, 2014.

[4] H. Xia, M. Dawande, and V. Mookerjee, "Role refinement in access control: model and analysis," INFORMS Journal on Computing vol.26, no.4, pp. 866-884, 2014.

[5] L. Bauer, L. F. Cranor, R. W. Reeder, M. K. Reiter, and K. Vaniea, "Real life challenges in access-control management," in Proceedings of the CHI Conference on Human Factors in Computing Systems, Association for Computing Machinery, pp. 899-908, 2009.

[6] D. E. Bell and L. J. LaPadula, "Secure computer systems: mathematical foundation report ESD-TR-73-275," MITRE Corp., 1973.

[7] Ponemon Institute, "2022 cost of insider threats global report," Proofpoint, 2022.

[8] D. Tsiostas et al., “The insider threat: reasons, effects and mitigation techniques,” in 24th Pan-Hellennic Conference on Informatics, pp.340-345, 2020.
[9] Y. Kodaka, H. Hasegawa, and H. Takakura, “A proposal for access control method based on file relation inference from users behavior(in Japanese),” IEICE Technical Report vol.123, no.86, pp. 40-47, 2023.
[10] T. Xu, H. M. Naing, L. Le and Y. Zho, “How do system system administrators resolve access-denied issues in the real world?,” in Proceedings of the CHI Conference on Human Factors in Computing Systems, pp. 348-361, 2017.
[11] M. Beckerle and L. A. Martucci, “Formal definitions for usable access control rule sets from goals to metrics,” in Proceendings of the Ninth Symposium on Usable Privacy and Security, pp. 1-11, 2013.
[12] M. L. Mazurek et al., “Exploring reactive access control,” in Proceedings of the CHI Conference on Human Factors in Computing Systems, Association for Computing Machinery, pp. 2085-2094, 2011.
[13] N. Shalev, I. Keidar, Y. Weinsberg. Y. Moatti, and E. B. Yehuda, “WatchIT: who watches your IT guy,” in Proceedings 26th Symposium on Operating Systems Principles, pp.515-530, 2017.
[14] Y. Desmedt and A. Shaghaghi, “Function-based access control (FBAC) from access control matrix to access control tensor,” From Database to Cyber Security, vol 11170, pp.143-165, 2018.