# Didactic Visualization for a Searchable Encryption Scheme

Ruilin Yang
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
r.yang-1@student.utwente.nl

## ABSTRACT

Searchable Encryption(SE) is an encryption technique that allows a user to delegate data storage to a third-party service provider, without compromising data confidentiality and searching functionality. The prevalence of cloud storage has given rise to the need for SE, and people who can work with SE in academics and applications. However, in general, the education of cryptography is difficult due to the complex nature of the subject. While it is common to use software systems for visual representations of algorithms for both teaching and laboratory exercises, there is no visualization tool for Searchable Encryption schemes yet. In this paper, we explore how visualization can help undergraduate CS students to understand an early SE scheme as well as the general idea of SE. The proposed visualization prototype is available at: `https://github.com/RuilinYang-beta/SearchableEncryption`.

## Keywords

Searchable Encryption, Visualization, Didactic Design, Education

## 1. INTRODUCTION

With the rise of Cloud Services, it is desirable to put data on a cloud storage service for individuals and organizations, due to its lower costs and better data accessibility than purchasing and maintaining the storage hardware locally. At its most basic level, a client sends his/her files to a cloud server, which records the information. When the client wishes to retrieve certain information, for example, based on a search term, the cloud server either sends back the related files or allows the client to access them on the cloud. This poses a question, how can the server tell which files are of interest to the client, based on the input search term? Since the data should be encrypted prior to outsourcing (otherwise, the client initiates the compromise of the confidentiality of his/her own data), a naive approach would be for the client to encrypt the search term in the same way that the files were encrypted before sent to the cloud, then the server performs a sequential search and returns all the files containing the encrypted search term. However, this approach is not secure — it is based on the assumption that the encryption is deterministic (the same keyword always results in the same encrypted output) as in ECB mode, we know that the deterministic encryption preserves the pattern in the plaintext and is prone to statistical attacks, as the middle picture in Figure 1. shows.


*35$^{th}$ Twente Student Conference on IT* Jul. 2$^{nd}$, 2021, Enschede, The Netherlands.

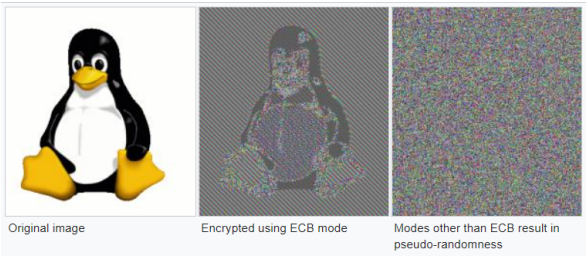




**Figure 1. Image encrypted in different modes. Taken from lecture slides of Peter, A., University of Twente(2018).**

Since in a secure scheme, the data should be encrypted non-deterministically, the client would not be able to know the ciphertext of a keyword, the client can send all the keys to the server for it to search on the (decrypted) data, however, this violates the purpose of encryption and leaves the data vulnerable to corrupted insiders at the server-side. Or, the server could just return everything to the client, but this is apparently inefficient and unfeasible [12].

To address this issue, Searchable Encryption schemes have been proposed since the early 2000s. Compared to the naive ideas above, searchable encryption involves more complex constructs and is harder for students to follow. There are already known difficulties in the education of cryptography in general: less solid mathematical knowledge of some students, limitation of class hours, and lack of practical exercises[2, 15]. Although it is a common practice to use visualization as an aid for algorithm-related education [16], and there has been evidence of improved teaching outcomes by adopting properly-designed educational software[2, 4, 20, 16], there is so far no visualization tool for searchable encryption. This research aims to design and implement a visualization to help students understand searchable encryption schemes, it asks the following research questions:

**RQ1: Which Searchable Encryption scheme(s) to design visualization for?** Searchable Encryption has expanded to a large field and several authors have systematically reviewed the field and categorized existing schemes. By scanning through the reviewing literature, we aim to identify the most feasible scheme to design the visualization for. The feasibility depends on the complexity of the scheme because it determines the time needed to understand it.

**RQ2: How to design and implement the scheme in a way**

**that helps novice learners learn?** This includes exploring existing cryptography education tools, reviewing guidelines of didactic visualization design, carrying out the design and implementation, and a pilot to evaluate if the prototype is helpful for the intended group.

The paper is structured as follows: Section 2 explores the existing educational tools; Section 3 synthesis the requirements for the prototype by reviewing the principles of didactic design; Section 4 describes the prototype in detail; Section 5 evaluates the usability and educational goals of the prototype; Section 6 briefly discusses the results of the evaluation; finally, Section 7 concludes the paper.

## 2. EXISTING SOLUTIONS

There already exist a few visualization tools in cryptography education. However, many of them are either not publicly available, or used to be publicly available but now lack apparent maintenance, such as the GRASP (GRaphical Aid for Security Protocols) [13] tool for students at the United States Air Force Academy, the GRACE (Graphical Representation and Animation for Cryptography Education) [4] tool for undergraduate students at University of Salerno, and the COALA (CryptOgraphic ALgorithm visuAl representation) [16] tool for students at University of Belgrade. Some more tools are unnamed and are even more untraceable [20, 10], apart from the screenshots of them that are kept in the corresponding papers.

There are two tools that are publicly available, achieve outstanding longevity, and act as the basis for many educational experiments [1, 7, 9, 19]: CrypTool 2 and JCrypTool [5]. Both headed by researchers at University of Siegen, the two tools are open source and support a wide range of operations. The differences between the two are the supported Operating System (Windows only for CrypTool 2, and all platforms for JCrypTool), the underlying technology (C#/.NET for CrypTool 2, and Java Eclipse RCP for JCrypTool), and the way a user can interact with the tool (users can connect building blocks to create his/her own cryptography systems in CrypTool 2, whereas in JCrypTool users can only follow the instructions in existing components). However, both platforms do not have built-in visualizations for searchable encryption, and in CrypTool 2, although users have the freedom to construct a cryptography pipeline, some operations in searchable encryption require an even higher degree of control such as splitting each fixed-sized chunk of a file into two parts and performing different operations on each part.

We see positive feedbacks from the author of the papers, for example, compare to the "chalkboard" approach, the visualization tools have the ability to make changes on the fly, demonstrate "what-if" cases for student discussion, and the "gee-whiz" moment is engaging that students tend to pay closer attention to what is happening [4]. It is attempting to build a tool that can have similar engaging effects to help students learn about searchable encryption.

## 3. REQUIREMENTS

Since the learning of cryptography can be intimidating for students due to the mathematically heavy knowledge background and a large number of terminologies [8], the design aims to help novice students gain hands-on experiences and develop an intuition without being bogged down by the mathematical and technical details, for this purpose a certain degree of abstraction is needed.

A study about Didactic Design [21] points out several principles that are suitable for the prototype: *the principle of conciseness* that the amount of information presented should be concise and meaningful to get the attention of the students; *the principle of autonomy* that grouping information blocks by their semantic load; *the principle of structure* that combining the reference points of logically related semantic blocks to the whole picture of the information, to help students understand not only each block but the logical relationships between them; *the principle of quality* that singling out the most important bits of information in terms of the observer's perception by creating visual anchors; *the principle of phasing* that presents the information in a controlled order to suit the logical flow of the teaching materials; *the principle of simplicity and accessibility* that considers the perceptibility of information and avoid overloading.

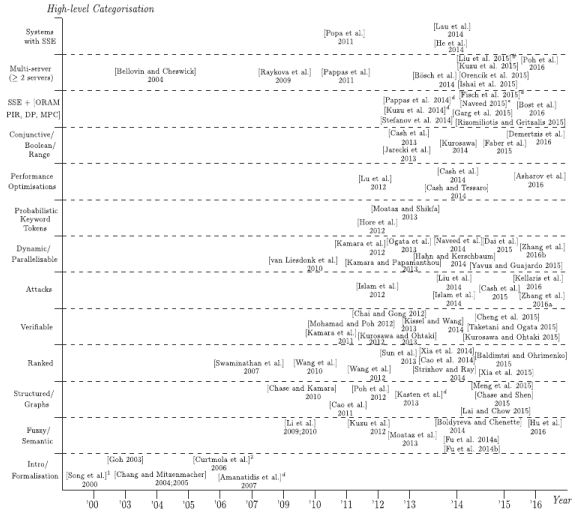



**Figure 2. Symmetric Searchable Encryption(SSE) timeline. Taken from [12]**

We adopt the aforementioned principles as qualitative guidelines of the product, of which the usability and educational functionality will be evaluated in a pilot test.

## 4. THE PROPOSED PROTOTYPE

This section first introduces the chosen scheme on a high level, then describes the proposed visualization prototype in detail.

### 4.1 The chosen scheme

All the reviewing literature categorizes Searchable Encryption schemes into Symmetric Searchable Encryption (SSE) and Public-key Encryption with Keyword Search (PEKS) [3, 18, 6, 11, 17], only one of them makes the categorization with minor variance [17]. Comparing SSE and PEKS, the former involves simpler constructs. A review [12] specifically for SSE presents a high-level categorization of the schemes. We choose the very first scheme by Song et al (2000) [14] to implement, as it relies on a few common cryptographic building blocks, a nice side effect is that for those who have been acquainted with them, the scheme offers a chance to brush them up; and for those who have not seen them, it illustrates how these primitives are being used in action.

The security of the scheme proposed by Song et al. relies on a few cryptographically secure primitives:

**Pseudo-random Generagor** $G$ that deterministically expand a short, uniformed seed into a longer pseudorandom

output that is indistinguishable from truly random bits;

**Pseudorandom keyed function** $F$ that for a uniform key $k \in \{0,1\}^n$, the function is indistinguishable from a uniform function;

**Pseudorandom keyed permutation** $E$ for a uniform key $k \in \{0,1\}^n$, the function is indistinguishable from a uniform permutation.

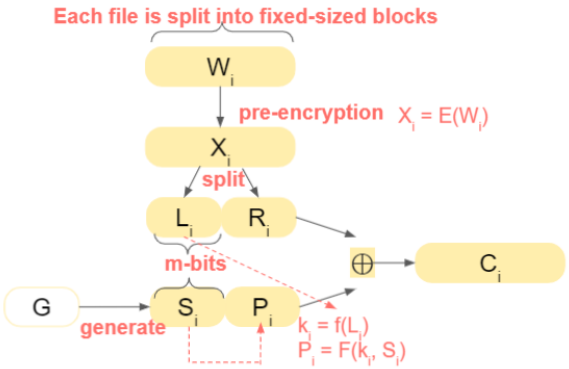



**Figure 3. The client encrypts a plaintext block $W_i$ into a cipherblock $C_i$ in a non-deterministic way. Taken from the embedded introductory slides of the proposed prototype.**

Here we provide a description of the scheme at an applied level, omitting many details. In the following paragraphs we use everything with a subscription $i$ to denote the data specific to the *i-th* block; use the $\oplus$ symbol to denote the XOR operation; use the $\langle a, b \rangle$ to denote the concatenation of $a$ and $b$.

First, at the client-side, each file is encrypted in a non-deterministic way before sending to the server (see Figure 3): each file is split into fixed-sized blocks $W_i$, each block then get pre-encrypted and split into two parts ( $E(W_i) = \langle L_i, R_i \rangle$ ), the left part $L_i$ is used to compute the key $k_i$ specifically for this block. The pseudorandom generator $G$ generates a sequence of pseudorandom bits, of which a specific chunk $S_i$ is applied to this block. $S_i$, together with the computed $k_i$, are fed to the pseudorandom function $F$, to produce the output $P_i$. The pre-encrypted block is then XORed with $\langle S_i, P_i \rangle$, where the result is the ciphertext $C_i$ of this block. The pseudorandomness in $S_i$ and $P_i$ masks the pattern in the plaintext.

At a later point (Figure 4), when the client wants to retrieve all the documents containing a certain search term, s/he do the same pre-encryption on the desired search term $W$, and compute the key $k$ in the same way when s/he first computes the ciphertext, finally sends $X$ together with $k$ to the server. Notice that it is the pre-encrypted word $X$ that is sent to the server, not the plaintext $W$, this design achieves *hidden query*, where the client search for a word without letting the server know the word in its plain form. It also achieves *controlled search*, as the $k$ can only enable the server to verify whether a ciphertext block is computed from the desired search term, and the server can learn nothing else, as we shall see in the next step.

Now we switch to the role of the server, after receiving the $X$ and $k$ from the client (Alice), the server can perform an XOR operation on the $X$ and each ciphertext block $C_j$ (Figure 5). Observe that, compared to the XOR step in Figure 3, the computation here swaps the XOR output

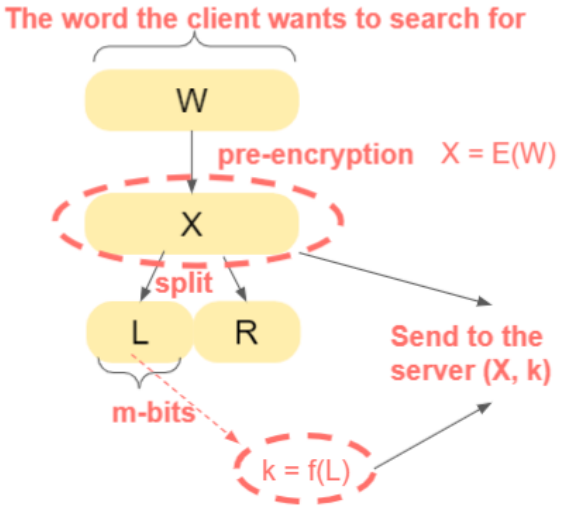



**Figure 4. The client submits the encrypted search term $W$ and the key $k$ to the server. Taken from the embedded introductory slides of the proposed prototype.**

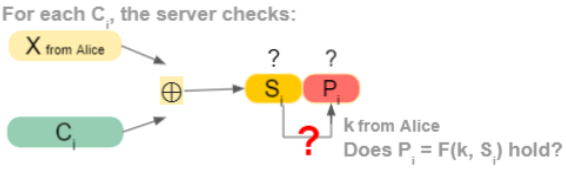



**Figure 5. The server verifies whether a ciphtertext block $C_j$ is computed from the given search term. Taken from the embedded introductory slides of the proposed prototype.**

and one of the inputs. If this $C_j$ is computed from the $X$ from the client, then the outputs are the original $S_i$ and $P_i$ used in the client-side encryption, and the relationship between them should be preserved; otherwise, there is no relationship between the left and right part of the output. The server can use this property to verify if a ciphertext block is indeed derived from the desired search term. In this way, the server can determine whether a file contains the desired search term, and decides whether to return the file accordingly.

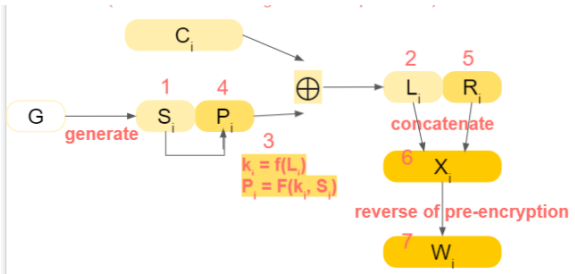



**Figure 6. The client decrypts a block of the returned files from the server. Taken from the embedded introductory slides of the proposed prototype.**

Finally, when the client receives the returned file from the server, s/he needs to recover them from ciphertext to plaintext (Figure 6). It is done as the following: first, the client re-produces the pseudorandom bits $S_i$ then use it to recover $L_i$ together with the left part of $C_i$; the recovered $L_i$ can be used to compute the key $k_i$, which can

be further used to compute $P_i$; finally, $R_i$ can be recovered from XORing the right part of $C_i$ and $P_i$, gluing $L_i$ and $R_i$ together and the plaintext $W_i$ can be recovered by reversing the pre-encryption process.

As we see, even the earliest (also one of the simplest) scheme takes some effort to explain. The reason could be due to the different actions between each party and the many intermediary products. For a novice learner, we can imagine that by just reading the paper, which only provides an image of the client's encryption part, it might be hard to comprehend how the search and recovery are done.

## 4.2 The proposed visualization prototype

We adopt the convention to call the first two parties involved in the scheme Alice and Bob. In our case, Alice is the client, and Bob is the cloud server.


**Figure 7. The two types of help information that are accessible on every page.**

First of all, it is assumed that the students who uses this prototype do not know the scheme by heart, in using the prototype, two types of information might be needed: the theoretical knowledge about the scheme, and the practical information on how to use the prototype. It would be too much load if we present these two pieces of information together with the operational user interface, so a welcome page is designed to inform the user, that whenever s/he is stuck, s/he can brush up the knowledge via the "intro" menu on the top-left corner, and practical instruction on the top-right corner. When the user clicks on either of them, a separate window pops up to keep the main window simple, see Figure 7.

Then, four pages representing the entire flow are designed. Here we only present the first page in this section. For the rest pages, please refer to Appendix A.

On the first page, the user acts as Alice, the actions are selecting files, setting a password, initializing the primitives, and performing the pre-encryption for all the files within one click, see Figure 8.

On the second page, the user still acts as Alice, s/he prepares all the necessary intermediary products and finally computes the ciphertext of all blocks across all the files (see Figure 12, Appendix A).

Between the second and third page, an optional animation shows up (it can be turned off on the first page for experienced users), reminding the novice user of the context (see Figure 13, Appendix A).

On the third page there is a split view: where the user acts as Alice first, and then Bob. When acting as Alice, the user input a search term in plaintext $W$ and computes the actual query terms $X$ and $k$ to send to Bob. When acting as Bob, the user performs a step-by-step computation and reaches the conclusion of whether a file contains the term Alice is looking for (see Figure 14, Appendix A). In between the role transition from Alice to Bob, and after Bob has decided the files to return, there are also optional animations that can be turned off on the first page.

On the fourth page, the user plays again the role of Alice to decrypt the returned files from Bob. There is a grey (inactive) image reminding the user how the ciphertext is made, and a normal (active) image illustrating the decryption process (see Figure 15, Appendix A).

Across all the pages, there are shared features that aim to assist learning: *the always-present image* in the first two pages are informing the user the ongoing process, and the last two pages each have two images (one for reminding the user an important past operation, one for current operation) helping the user to stay on the track; *the hover-highlight effect* of text area that highlights related primitives and/or components in the image whenever the user hover on a text area (as shown in Figure 12, Appendix A) aims to help the user connect bits of information to the overall process; *the hover-highlight effect* of the primitives (as in Figure 8) shows the key for each primitive, this is aimed at letting the user perceive a certain degree of transparency; the same aim goes for *the file-clicking effect*, when the user clicks on a filename, all the text area updates to the relevant content of that file; finally, *the order of button click* is strictly controlled, in a way that certain values can only be computed after some other values are computed, and the "next" button can only be clicked when all the value on the screen is computed.

This design connects to the requirements in the following ways: *the always-present big picture* and *the hover-highlight effect* of the text area connect to *the principle of structure* and *the principle of quality*, for that they aim to help students see the logical relationship between a single operation and the entire process in an attention-grabbing way; *the hover-highlight effect* of the primitives and *the file-clicking effect* connect to *the principle of simplicity and accessibility*, for that information is hidden but can be easily retrieved within a mouse hover or a click; *the order of button click* connects to *the principle of phasing*, for that it guides the student in an implicit way; and finally, the overall design connects to *the principle of conciseness* for each page contains nothing else than the essentials, and to *the principle of autonomy* for the content on each page is semantically close.

# 5. EVALUATION OF THE PROTOTYPE

To evaluate whether the proposed visualization prototype is easy to use, and is helpful for students to understand the cryptography scheme, a small-scale pilot is conducted. This section covers the demography of the group, the evaluation process, and the evaluation results.

## 5.1 Demography and evaluation process

This prototype is intended for undergraduate students in computer science. In total, 5 participants are recruited, all of whom have obtained 45-165EC. All of the participants did not have dedicated cryptography courses in their curriculum so far.

The process goes as follows: first is the preparation phase, where a participant downloads the prototype on his/her computer and reads the instructions, which includes a brief description of the process and three hints on where to get help within the prototype; then the participant opens the prototype and reads the introductory slides of Searchable Encryption and the chosen scheme, then carry out the experiment with the scheme; finally, the participant answers seven questions relating to his/her experience, the question list can be found in Section 5.2.

The participants do this experiment one by one, as in this way we can carefully observe how each of them use the

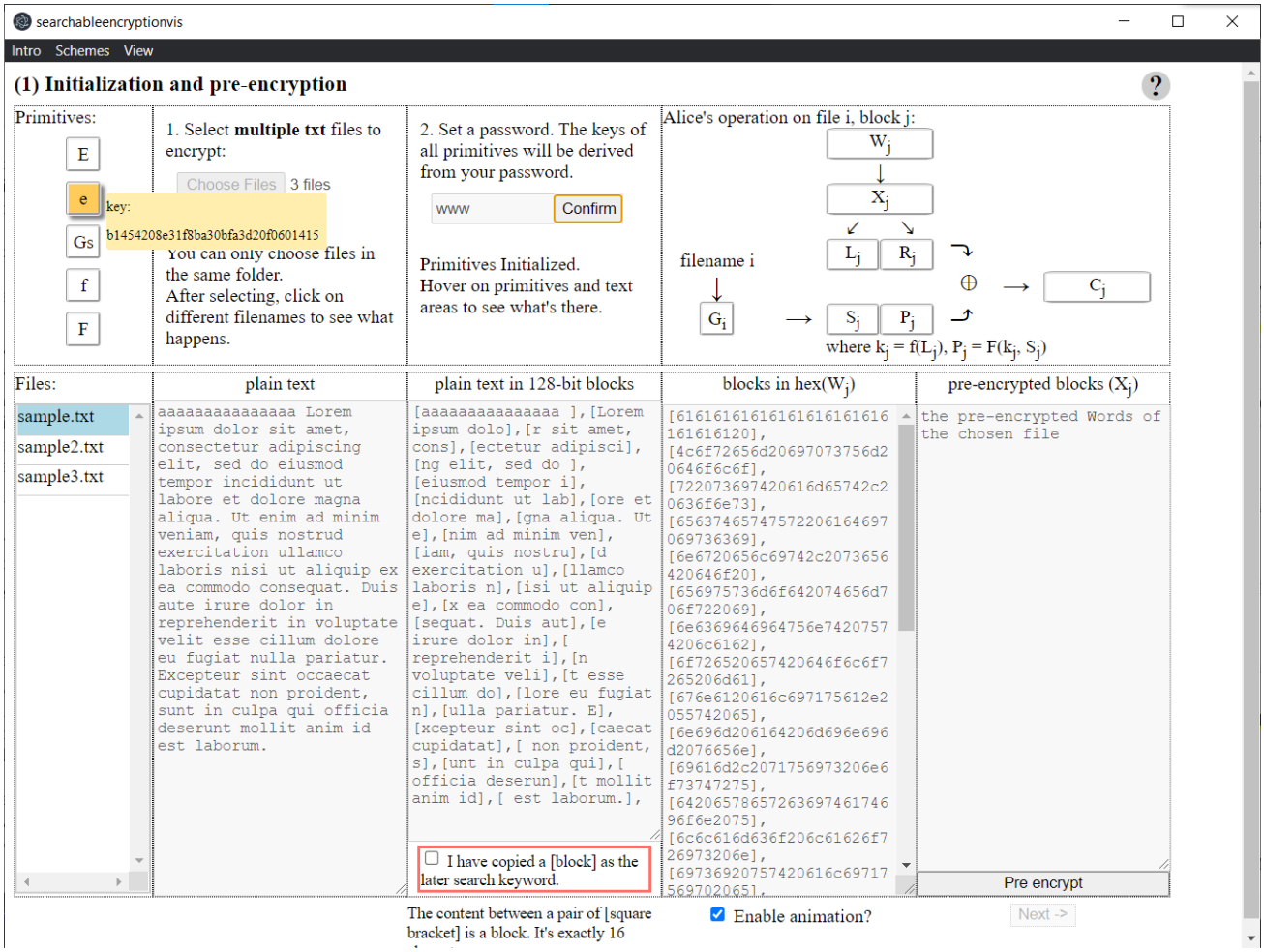



**Figure 8. The first page: choose files, initialize the primitives, and pre-encrypt all files.**

prototype. The entire process of each participant is under our (mostly silent) observation, the goal of the observation is to see the nuances in using the prototype, such as where do participants get stuck, and whether they can recover from the stuck on their own, as these subtleties might not necessarily be reflected in the follow-up questionnaire, but might be helpful in identifying where to improve the prototype. In case the participant gets stuck and could not get through for a while, we talk to him/her to provide help.

## 5.2 Evaluation results

For convenience, the 5 participants are called A, B, C, D, and E in this section, named after the chronological order they conduct the experiment. Participant A experienced the first version of the introductory slides which led to turbulence in his experience as we shall see. We adjusted the slides and kept them the same for participant B, C, D, and E. Because participant A's response still sheds some light on how to design an educational visualization, we keep his records here, but won't aggregate his data with the rest of the participants.

It is expected that a participant can finish the whole process in 30 to 60 minutes, but as we see in Figure 9, participant A took a significantly longer time to complete, and is the only participant who requires further discussion about the scheme.

From the real-time observation, we can clearly see this

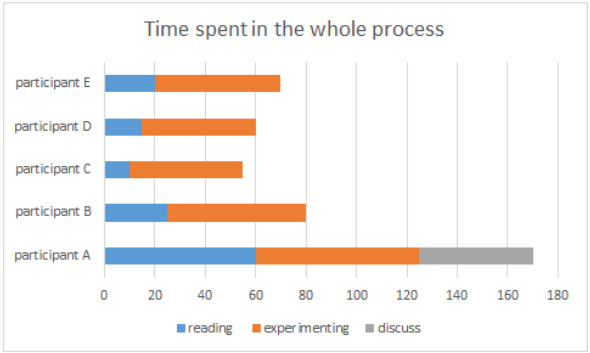



**Figure 9. The time each participant took for the whole process.**

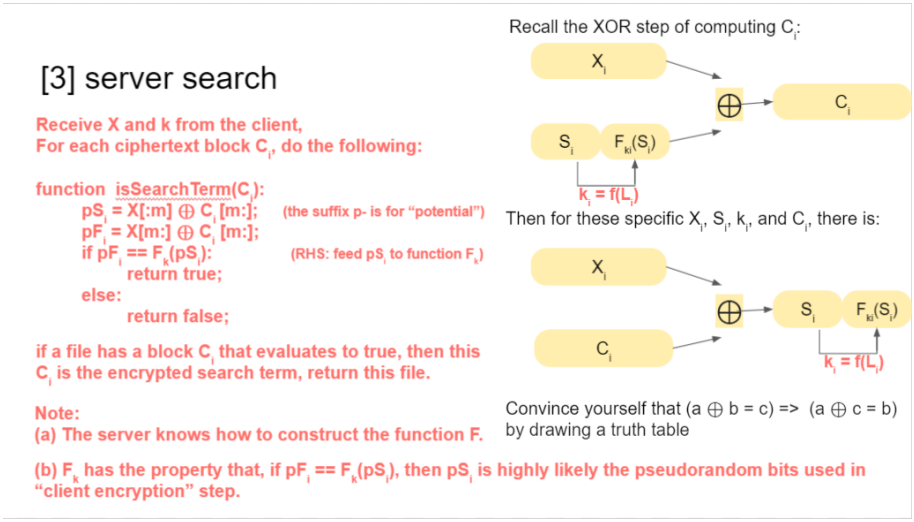
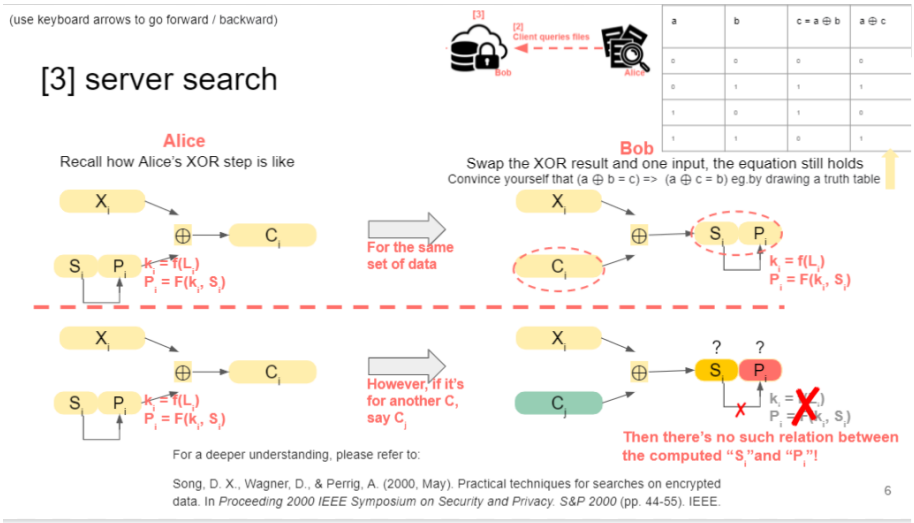



**Figure 10. The most confusing slide for participant A.**



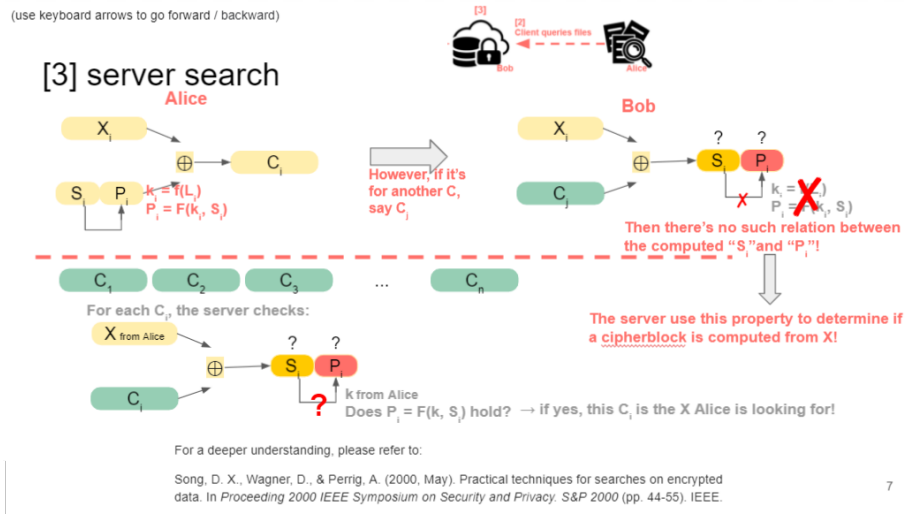




**Figure 11. The previously most confusing slide is split into two slides.**

is because the introductory slides are overly complicated, for example, the introduction slides to the scheme contains the pseudocode of each step (see Figure 10). The intention was to make the material as self-contained as possible, but it turned out that the details unnecessary for the first encounter would trap the participant's attention and made him strained because he thought that "every bit of information should be understood in order to perform the experiment", and the strains later affect the experience in experimenting, as he said, "the slides contain overwhelming information, after reading, I still can't put the pieces together and connect to the steps in the experiments, so when the 'next' button is grey, I have no idea what to do and get frustrated".

His feedback is crucial, it is assumed a few adjusting of the way the information is presented can largely improve the participants' experience, so we improved the introductory slides of the scheme in the following ways: the textual elements are replaced with visual elements(solid arrows, dashed arrows, colors, etc) as much as possible; optional information that is not necessary are marked with "for interested readers"; a reference to the paper is sticked to the bottom of each of the introductory slides to hint that the reader doesn't have to understand it in much depth for their first encounter with the prototype. For an example of these changes, see Figure 11, where the previously most confusing slide is modified into two illustrative slides.

**Table 1. The results of Q1 through Q4. The 4th column is the averange score of participant B,C,D,E and the 5th column is the standard deviation.**

| no. | question | A | avg of B,C,D,E | std of B,C,D,E |
|---|---|---|---|---|
| Q1 | How easy is the prototype to use? 1 for very easy, 5 for very hard. | 1 | 1.75 | 0.96 |
| Q2 | How helpful are the slides under the "intro" menu? 1 for not at all, 5 for very helpful. | 1 | 4 | 0.71 |
| Q3 | How helpful is the "?" button on each page? 1 for not at all, 5 for very helpful. if not used, you can skip this question. | 4 | - | - |
| Q4 | How confident are you to learn the scheme in more depth? 1 for not at all, 5 for very confident | 1 | 3.5 | 0.87 |

Participant B, C, D, and E who came later read the improved version of the slides. Table 1 is the result of questions 1 through 4. A's answers to Q2, Q3, and Q4 differs a lot from those of B, C, D, and E; in addition, A is the only one to check out the "?" button on each page while experimenting, though this provides some technical aid that A found helpful, it did not help him to comprehend what is going on and he ended up frustrated and has no confidence in further learning at all.

In contrast, participant B, C, D, and E highly value the improved introductory slides as shown in their answers to Q2; in practice, it is observed that three out of the four are

**Table 2. The results of Q5 through Q7. The last column is the number of participant(s) who mentioned a certain point.**

| no. | question | answers | count |
|---|---|---|---|
| Q5 | What is the point(s) you like the best about the app? | + highlight effect | 2 |
| | | + slides look good | 2 |
| | | + slides are informative | 1 |
| | | + coupling of slides and experiment. | 1 |
| Q6 | What is the point(s) you like the least about the app? | - there's no "back" button | 2 |
| | | - instruction to copy a "block" is not clear enough | 2 |
| | | - the page is not responsive to shrinking the window | 1 |
| | | - cannot put introductory slides and experiment slide by side | 1 |
| | | - sample files are not easily accessible | 1 |
| | | - some text is squeezed into the neighboring cell | 1 |
| Q7 | What is your suggestion to improve the app? | clearer instruction to copy a blockn | 2 |
| | | "back" button | 2 |
| | | inform users it is not necessary to understand everything on the slides all at once | 1 |
| | | an easy way to access sample files | 1 |
| | | additional information when hover over the image | 1 |

making effort to link each small piece of operation to the bigger picture: participate B tried to experiment a little bit, then go back to the slides on the experimented part to review it, and then go further experimenting a little bit and review the slides a little bit more, and so on; participant C and D carefully examined the highlight effect on the image when they hover on each corresponding text areas, their mouse stopped a while, which looks like an internalizing thinking process is going on, as C later comments "the context and motivation, as well as the whole framework are clear". Although progress is made in understanding the big picture, participants B, C, D, and E tend to stay neutral in Q4, because "it is not clear what still needs to be learned" (quote participant C) and "a lot more effort must be needed" (quote participant E).

Question 5 through 7 are qualitative, and the result is encoded in Table 2. Participant A's answer is not included in the table because he experienced an older version of the slides, and this affects all his following reactions. We see the slides are valued as among the most prominent virtues of the prototype; the hover-highlight effect and the coupling of the slides and the hands-on experiment are also well-received.

The major weaknesses of the prototype are shown in the answers to Q6 and Q7, they can be categorized into two types: (pure) UI-related or education-UI category. The (pure) UI category includes an unintuitive tip on copying a block; lack of a "back" button to go to the previous pages; lack of responsive design of the page; some text un-

expectedly squeezed into neighboring cells on some participants' computers; and the tortuous way to find sample files. The education-UI category includes the participants can be better informed that it is not necessary to understand the slide all at once, and that when hovering over the image it is expected that more information would show up (although more information is accessible through the "?" button, participant B, C, D, E didn't think of clicking on it).

## 6. DISCUSSION

The purpose of the proposed visualization, corresponding to the requirements, is twofold: one for being educative, and the other for providing a smooth user experience such that the student can easily use it by themselves. Among the seven questions asked in the evaluation, Q2 and Q4 are intended to test the first goal; Q1 and Q3 are to test the second goal; Q5, Q6, and Q7 are open questions that might induce answers related to either goal or to some other aspects other than the goals.

From the answers to Q2, Q4, and Q5 through Q7, we can see the educative goal is relatively well-received, with positive comments on the design of the slide, and the coupling of educative slides and hands-on experiments. From the answers to Q1, Q3, and Q5 through Q7, we can see that there are still various aspects of usability that can be improved.

One interesting observation from the discrepancy between the experience of participant A and the rest of the participants is that ill-received educational information correlates to the poor practical experience, and vice versa. In the case of participant A, although the technical aid via the "?" button served its purpose, it did not save participant A from ending up frustrated. But of course, due to the small number of participants reading the old/new version of the slides, whether the correlation holds true on a larger scale is to be verified.

## 7. CONCLUSION

In summary, this research first identifies one SE scheme to design a visualization prototype for because of its relative simplicity; then explores the scarcity of educational software in cryptography and the vacuum for the visualization of SE schemes; with the guidance of didactic design principles, a prototype has been developed. Five students of the intended group evaluated the prototype, though the first student's drastically long evaluating time leads to an overhaul of the introductory slides, there are still four effective evaluators of the improved version; judging from the results of the evaluation questions, the educational goal can be deemed as met, but the usability of the prototype has ample space of improvement.

Here we have seen the approach of presenting an visualization applet that the student can play around with, together with high-level introductory slides helped the students' first encounter with the earliest Searchable Encryption scheme, does the same approach apply to other Searchable Encryption schemes? Most other schemes let the client generate a searchable encrypted index that the server later search on, rather than search directly on ciphertext. Whether the same visualization approach apply to them can be further studied.

## 8. ACKNOWLEDGMENTS

I would like to thank my supervisor Dr.ing Florian Hahn, it is a pleasure to work with him, without his insightful guidance and warm encouragement, this research would not have been finished on time.

.

## 9. REFERENCES

[1] S. Adamović, I. Branović, D. Živković, V. Tomašević, and M. Milosavljević. Teaching interactive cryptography: the case for cryptool. In *IEEE Conference, ICEST*, 2011.

[2] S. Adamovic, M. Sarac, D. Stamenkovic, and D. Radovanovic. The importance of the using software tools for learning modern cryptography. *International Journal of Engineering Education*, 34(1):256–262, 2018.

[3] C. Bösch, P. Hartel, W. Jonker, and A. Peter. A survey of provably secure searchable encryption. *ACM Computing Surveys (CSUR)*, 47(2):1–51, 2014.

[4] G. Cattaneo, A. De Santis, and U. F. Petrillo. Visualization of cryptographic protocols with grace. *Journal of Visual Languages & Computing*, 19(2):258–290, 2008.

[5] B. Esslinger. Cryptool–an open source project in practice. *Lessons learned from a successful open source project. Published in Datenschutz and Datensicherheit*, 2009.

[6] F. Han, J. Qin, and J. Hu. Secure searches in the cloud: A survey. *Future Generation Computer Systems*, 62:66–75, 2016.

[7] S. Hick, B. Esslinger, and A. Wacker. Reducing the complexity of understanding cryptology using cryptool. In *10th International Conference on Education and Information Systems, Technologies and Applications (EISTA 2012), Orlando, Florida, USA*, 2012.

[8] Y. Kurt Peker. Modules for integrating cryptography in introductory cs and computer security courses. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, pages 738–738, 2017.

[9] M. K. Loussios. Cryptool 2 in teaching cryptography. *Journal of Computations & Modelling*, 4(1):349–358, 2014.

[10] M. A. Mayouf and Z. Shukur. Features of a visualization tool for specification and analysis of security protocol. In *2008 International Symposium*

*on Information Technology*, volume 4, pages 1–5. IEEE, 2008.

[11] H. Pham, J. Woodworth, and M. Amini Salehi. Survey on secure search over encrypted data on the cloud. *Concurrency and Computation: Practice and Experience*, 31(17):e5284, 2019.

[12] G. S. Poh, J.-J. Chin, W.-C. Yau, K.-K. R. Choo, and M. S. Mohamad. Searchable symmetric encryption: designs and challenges. *ACM Computing Surveys (CSUR)*, 50(3):1–37, 2017.

[13] D. Schweitzer, L. Baird, M. Collins, W. Brown, and M. Sherman. Grasp: A visualization tool for teaching security protocols. In *Proceedings of the 10th Colloquium for Information Systems Security Education*, volume 4, 2006.

[14] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pages 44–55. IEEE, 2000.

[15] X. Song and H. Deng. Taking flexible and diverse approaches to get undergraduate students interested in cryptography course. In *2009 First International Workshop on Education Technology and Computer Science*, volume 2, pages 490–494. IEEE, 2009.

[16] Z. Stanisavljevic, J. Stanisavljevic, P. Vuletic, and Z. Jovanovic. Coala-system for visual representation of cryptography algorithms. *IEEE Transactions on Learning Technologies*, 7(2):178–190, 2014.

[17] U. Varri, S. Pasupuleti, and K. Kadambari. A scoping review of searchable encryption schemes in cloud computing: taxonomy, methods, and recent developments. *The Journal of Supercomputing*, 76(4):3013–3042, 2020.

[18] Y. Wang, J. Wang, and X. Chen. Secure searchable encryption: a survey. *Journal of communications and information networks*, 1(4):52–65, 2016.

[19] R. Yang, L. Wallace, and I. Burchett. Teaching cryptology at all levels using cryptool. In *Proc of the 15th Colloquium for Information Systems Security Education Fairborn*, pages 13–15, 2011.

[20] X. Yuan, P. Vega, Y. Qadah, R. Archer, H. Yu, and J. Xu. Visualization tools for teaching computer security. *ACM Transactions on Computing Education (TOCE)*, 9(4):1–28, 2010.

[21] N. Zhytienova. Principles of visualization as a base of didactic design. *ScienceRise: Pedagogical Education*, (3 (11)):11–14, 2017.

# APPENDIX

## A. SCREENSHOTS OF PAGE 2, 3, AND 4 OF THE PROTOTYPE

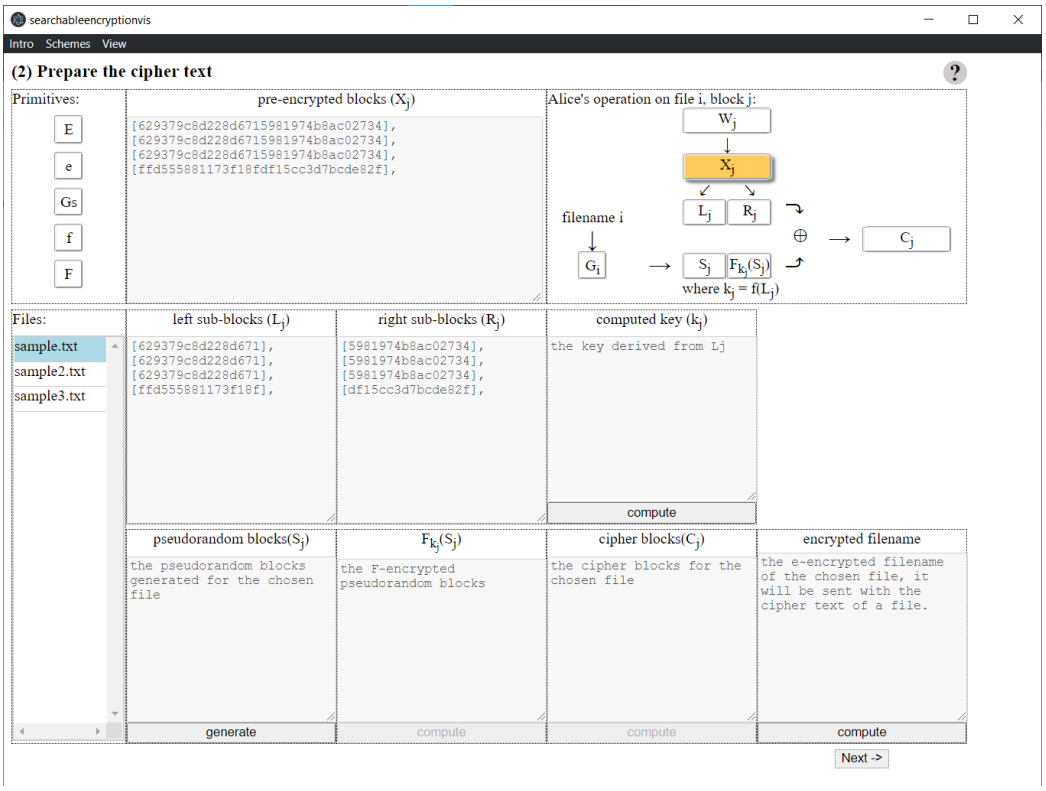



**Figure 12. The second page: Alice prepares the ciphertext across all the files. The picture is taken when the mouse is hovering on the $X_j$ text area, as a result, the $X_j$ in the picture on the top-right corner is highlighted.**


**Figure 13. The optional transition animation between the second and third page**

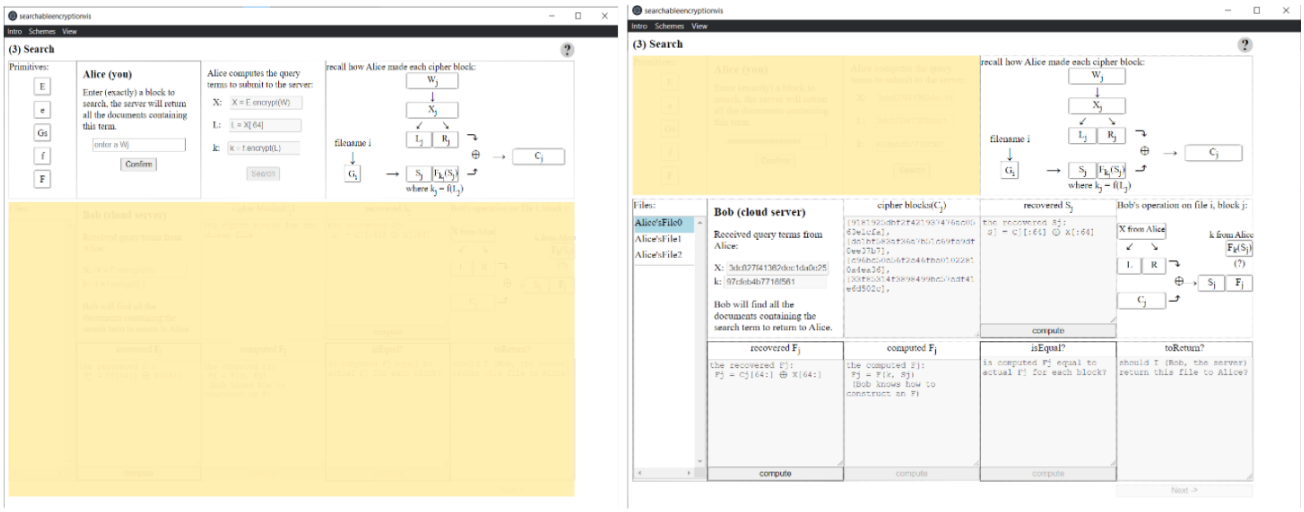


**Figure 14. The third page: the split view where the upper part is Alice's perspective, and the lower part is Bob's. The student first acts as Alice, then Bob. There is an optional animation that Alice sends query terms to Bob in between the role transition.**

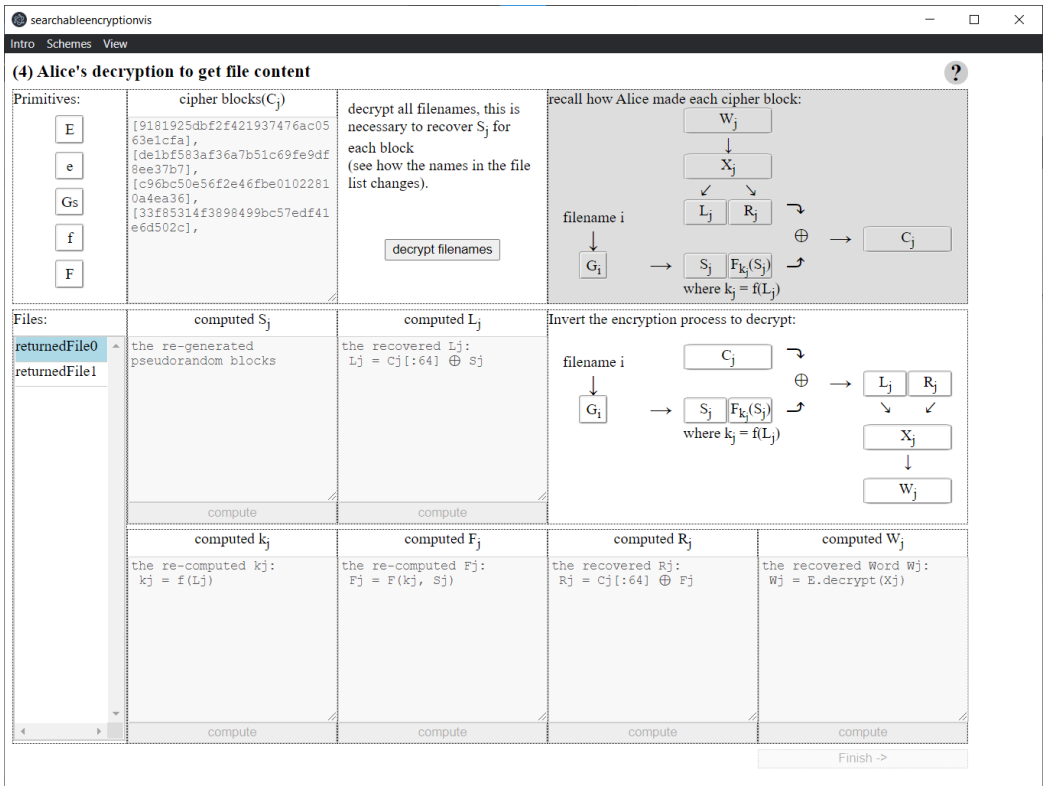


**Figure 15. The fourth page: where Alice decrypts the returned files from Bob.**