# A Usability Evaluation of Let's Encrypt and Certbot: Usable Security Done Right

Christian Tiefenau
University of Bonn
Bonn, Germany
tiefenau@cs.uni-bonn.de

Emanuel von Zezschwitz
University of Bonn & Fraunhofer FKIE
Bonn, Germany
zezschwitz@cs.uni-bonn.de

Maximilian Häring
Fraunhofer FKIE
Bonn, Germany
haering@cs.uni-bonn.de

Katharina Krombholz
CISPA Helmholtz Center
Saarbrücken, Germany
krombholz@cispa.saarland

Matthew Smith
University of Bonn & Fraunhofer FKIE
Bonn, Germany
smith@cs.uni-bonn.de

## ABSTRACT

The correct configuration of HTTPS is a complex set of tasks, which many administrators have struggled with in the past. *Let's Encrypt* and Electronic Frontier Foundation's *Certbot* aim to improve the TLS ecosystem by offering free trusted certificates (Let's Encrypt) and by providing user-friendly support to configure and harden TLS (Certbot). Although adoption rates have increased, to date, there has been only a little scientific evidence of the actual usability and security benefits of this semi-automated approach. Therefore, we conducted a randomized control trial to evaluate the usability of Let's Encrypt and Certbot in comparison to the traditional certificate authority approach. We performed a within-subjects lab study with 31 participants. The study sheds light on the security and usability enhancements that Let's Encrypt and Certbot provide. We highlight how usability improvements aimed at administrators can have a large impact on security and discuss takeaways for *Certbot* and other security-related tasks that experts struggle with.

## CCS CONCEPTS

•**Security and privacy** →**Usability in security and privacy**; *Web protocol security*.

## KEYWORDS

Usable Security; Let's Encrypt; HTTPS

**ACM Reference Format:**
Christian Tiefenau, Emanuel von Zezschwitz, Maximilian Häring, Katharina Krombholz, andMatthew Smith. 2019. A Usability Evaluation of Let's Encrypt and Certbot: Usable Security Done Right. In *2019 ACM SIGSAC Conference on Computer & Communications Security (CCS '19), November 11–15, 2019, London, United Kingdom.*ACM, New York, NY, USA, 18pages. https://doi.org/10.1145/3319535.3363220


*CCS '19, November 11–15, 2019, London, United Kingdom*

ACMISBN978-1-4503-6747-9/19/11.
https://doi.org/10.1145/3319535.3363220

## 1 INTRODUCTION

Transport Layer Security (TLS) is among the most important protocols to secure data in transit, and has been an active research topic in the usable security domain, especially regarding the end-user's perspective, e.g., [20,39,44]. For a decade, substantial effort has been invested in improving the efficacy of TLS warnings. From one of the earliest works by Sunshine et al. [44] to today, usable security researchers have attempted to find ways to help end-users make good decisions when faced with such warnings.

However, end-users are only one part of the picture. Akhawe et al. conducted a large-scale measurement study [5] and estimated that end-users would see 15,400 false positive warnings per true positive warning due to server misconfigurations.

In 2015, Let's Encrypt (LE) began operating, to increase TLS adoption by offering free certificates. Let's Encrypt is a non-profit certificate authority (CA) that was founded "to reduce financial, technological, and education barriers to secure communication over the Internet" [3]. In conjunction with LE, the Electronic Frontier Foundation (EFF) offers Certbot, a tool that automates the acquisition and configuration of LE certificates for web servers [18]. The hope of this initiative is to reduce the barriers and improve the usability of the TLS setup. The data published by LE suggests that adoption rates are rising [32], and that it is mainly impacting the lower-cost end of the web, as 98% of the LE certificates are issued for domains outside the Alexa 1M [4].

Manousis et al. [34] found that only 50% of the domains that obtained an LE certificate actually responded with a valid LE certificate on the standard HTTPS port. The authors concluded that despite the many positive effects of LE, "there are serious misconfigurations among many website owners who use Let's Encrypt".

To shed light on where the adoption problems above stem from, and to examine the advantages of LE, we conducted a **randomized control trial to compare the usability of the EFF's Certbot with the traditional certificate configuration approach**. The contributions of this paper are as follows:

(1) We present a quantitative study with 31 computer science students that compares the usability of two different methods for interacting with a certificate authority (CA) and configuring TLS on a web server.

(2) We show that Certbot's usability improvements are particularly important for lower-skilled participants.
(3) We analyze in which areas the automation of Certbot is particularly important.
(4) We discuss what lessons can be learned from Certbot and identify areas where these do not apply easily.
(5) We provide a methodological discussion of conducting lengthy laboratory user studies with expert users, such as administrators, and share the lessons learned.

## 2 RELATED WORK

When correctly deployed, TLS [12] protects the integrity and privacy of digital communication. However, different TLS features and protocol versions have been shown to have vulnerabilities, thus making several configurations (i.e., combinations of such features) insecure [11]. BEAST and DROWN are examples of effective and practicable attacks against TLS [8,26]. To understand the real-world vulnerabilities of the TLS ecosystem and the diversity of TLS (mis-)configurations, researchers examined TLS deployments in measurement studies and user studies.

### 2.1 Measurement Studies

Internet-wide scanning tools, such as ZMap [16] and Censys [13], are used to measure TLS in the wild. They were used in studies that identified frequent configuration problems that potentially lead to browser warnings and create attack surfaces [2,10,14].

Ouvrier et al. [38] passively monitored 232 million HTTPS sessions and reported that more than 25% of the sessions had weak security properties. Gustafsson et al. [23] analyzed differences in public Certificate Transparency (CT) logs, while Holz et al. [25] evaluated the security of email and chat infrastructures, and reported a worryingly high number of poorly secured servers . With the recent evolution of smart environments, new TLS-secured device classes have popped up. Samarasinghe and Mannan [41] measured the TLS parameters of 299,858 devices (e.g., cameras), and the authors found that such devices are usually more vulnerable than the Alexa Top Million sites. Common security problems included the use of RSA 512-bit keys, the RC4 stream cipher, or SSLv2 and SSLv3. Finally, Van der Sloot et al. [45] compared different measurement approaches and found that comparative analyses using aggregated CT logs, Censys snapshots, and Alexa 1M scans provide accurate snapshots of the TLS ecosystem.

Durumeric et al. [15] tracked the vulnerable population after the disclosure of Heartbleed, and found that, even after two days, 11% of the Alexa 1M sites remained vulnerable. Popular sites responded more quickly, while 3% of the analyzed population remained vulnerable as long as two months after being notified.

Kranch and Bonneau [28] investigated the use of novel security features such as HTTP Strict Transport Security (HSTS) and public-key pinning, and identified usability problems as the main reasons for reluctant upgrade behavior. The authors reported that *even conceptually simple security upgrades [are] challenging to deploy in practice.* Amann et al. [7] claimed that only the Signaling Cipher Suite Value (SCSV) and Certificate Transparency *have gained enough momentum to improve the overall security of HTTPS.*.

### 2.2 User Studies

Most TLS-related user studies focus on end-users, and their reactions to warnings. Sunshine et al. [44] conducted the first lab study examining the efficacy of current browsers' TLS warnings and evaluating two custom warning designs. Harbach et al. [24] studied how aspects of a warning message influence user reactions and found that linguistic properties have a strong impact. Several other studies were performed in the lab [42], online [20], and in the field [20,21] to analyze the impact of the warning design and contextual factors [39] on users' click-through rates, and found that better warning designs can increase adherence rates [20].

Compared to the wealth of research focusing on end-users, there is far less focused on administrators. Fahl et al. [19] surveyed 755 web developers and investigated the reasons for deploying non-validating X.509 certificates on publicly available websites. Although one third of the participants admitted to misconfiguring the web servers accidentally, the majority stated that they knew about the problem, and gave reasons for their configuration choices. For example, some system administrators mentioned the high prices of CAs as a reason for intentionally deploying non-validating certificates; others stated that they did not trust CAs or had trouble configuring virtual hosts. Based on a mental model study by Krombholz et al. [30], administrators lack of conceptual mental models of HTTPS.

Schechter et al. [43] conducted user studies where the authors compared the effect of role-playing in studies on the outcome. They showed in a phishing study with end-users that participants in the role-playing scenario behaved significantly less secure than those who faced a more realistic one. Komanduri et al. [27] also compared a survey to a scenario-based task description and found that users tended to choose better passwords in the latter scenario.

Most relevant to our research is Krombholz et al.'s [31] user study on the deployment process of HTTPS. The present study is an extension of their study protocol. They conducted an observational lab study with 28 knowledgeable users which simulated a simplified certificate acquisition and standard deployment process. The study used a minimal web-based CA where participants could acquire TLS certificates to be manually installed on an Apache web server. The study revealed a host of usability issues that often resulted in vulnerable configurations. The study did not contain conditions in which participants used LE and Certbot. The study also did not inform participants about which security requirements they should meet. The present study differs in several ways. First, we conducted a randomized control trial to compare a traditional CA approach to Let's Encrypt and Certbot. We also explicitly told participants which security goals should be reached, and how the security of the resulting configuration could be evaluated. We made this change because Naiakshina et al. found that computer science students did not add security unless explicitly asked to [36]. The final important difference in the study design is that we formalized the interaction between the experimenter and the participants. In the Krombholz et al. study, technical assistance was given; however, this was done in situ, and was not planned in advance. In addition, the help was not recorded, and it was not analyzed. We created a Mattermost support channel for in-study realism, as well as to deliver consistent and recorded interaction with the participants. Our records on when

participants required which kind of help o er valuable insights into the usability challenges. A nal important di erence concerns the participant sample. Krombholz et al. invited the 30 best students of the pre-screening survey of whom 28 participated in the study. We did not lter out lower-skilled participants because we wanted to see the e ects of Certbot on di erent skill levels.

In parallel research, Bernhard et al.'s work [9] analyzed the usability of Let's Encrypt in comparison to a traditional CA approach. They conducted two studies: one within subjects with nine participants and one between subjects with ten participants ( ve per condition). In the rst study, none of the nine participants managed to complete the traditional CA task, and only four managed to complete it with Let's Encrypt. In the second study, the authors got con icting information with three of ve managed to complete it in each condition. The authors stated that this was likely due to a change in recruitment criteria which was introduced in the second study to raise the skill level of the participants. Due to this, and the small sample sizes, the authors stated that they had found no reliable e ects, and even con icting information on which system o ers better usability. In conclusion, they wrote: *However, we did not nd conclusive evidence regarding which method [Let's Encrypt vs. Traditional CA] is more satisfactory to users, which enables more secure con gurations, which system users were more con dent in, nor which systems users would recommend. This is likely due to our small sample size, and future work is needed to better understand these features.* The present study has a larger sample size, so it does not su er from these issues. The study also gathered additional details via logging and the Mattermost support channel, so the analysis can go into more detail about where participants faced challenges and how Certbot helped them.

## 3 RESEARCH QUESTIONS

The research questions are split into two groups. The rst relate to the main subject matter, the usability of Certbot.

> Does Certbot support its users in ful lling the task of enabling TLS?
> Related work has shown that users struggle with manually deploying SSL certi cates. We want to measure Certbot's performance and capability to help administrators set up TLS correctly compared to the manual approach, to quantify the performance, as well as to draw lessons learned from the Certbot approach.
> How do participants perceive Certbot's functionality and usability?
> Although automated con guration has many usability bene ts, it is an open question whether administrators feel comfortable with the decreased level of control they might perceive due to automation.
> How can the Certbot process be improved?
> Although Certbot has a reputation for good usability, we are interested in possible areas of improvement, to support even more users in deploying secure TLS correctly. Because the usability is likely to be good from the start, we do not expect major improvements, but are open to the possibility.

The second group of questions relates to study methodology for administrator studies. Usable security researchers have a decade of experience in end-user studies. Studies with developers and administrators do not have the same body of knowledge yet. Naiakshina et al. found that the way tasks are framed for computer science students and freelance developers has a signi cant e ect on how participants deal with security [35 37]. To add to this body of knowledge, we introduce the following research question:

> How does task framing a ect how participants behave in the study?
> A common method used to elicit realistic behavior in end-user studies is to use a role-playing scenario [27 43]. We are interested in seeing whether this tool is also useful for studies with experts like administrators or developers who are represented in this study through student proxies [36].

## 4 METHODOLOGY

### 4.1 Study Design

Similar to Krombholz et al. [31], we opted for a lab study to monitor and control the participants' behavior. In contrast to Krombholz et al.'s study, we looked at two independent variables. We conducted an A/B test to compare the usability of Certbot with a traditional CA. Thus, we had two treatment conditions: CA-Certbot (CA-Cbot) for the Certbot with Let's Encrypt condition and CA-Traditional (CA-Trad.) for the traditional manual CA approach. Although it would have been nice if we could have used the same web CA as used by Krombholz et al. to enable a more direct comparison with their work, we opted to use a more complex one that resembles the realistic work ow of acquiring a certi cate from an existing CA. In particular, the method included ownership veri cation. There, a server owner has to prove that they are in possession of the server and domain by placing a speci c le in the web folder or by responding with de ned content to a request made by the CA. We opted for these improvements because they would give a fairer comparison for the CA-Certbot condition which has the full complexity of the real-world implementation. Because we assume that the con guration task is highly dependent on personal skills, we opted to study the two conditions within subjects, because the sample size which would have been needed to balance out personal skill in a between-subject design would have been unattainably huge. To counter learning and fatigue e ects, we randomized the order of conditions: Half the participants were assigned to use CA-Traditional rst, and the other half started with CA-Certbot rst.

The second variable is a meta-variable concerning the task framing. In a developer study conducted with students, Naiakshina et al. reported that in post-task interviews, some participants excused poor or no security performance by stating that they would have tried harder if they had been working for a real company as opposed to participating in a study [36]. This is a general problem for security-focused user studies in which participants know they are taking part in a study. There is always the risk that participants behave less securely because they know they are safe in a study environment or that they behave more securely because they want to impress the experimenters. A possible approach to mitigate this problem in end-user studies is to construct a role-playing scenario, and make the task as realistic as possible, to get participants into the right frame of mind. However, because we do not have the

body of experience with expert studies that we do with end-users, it is not clear whether this kind of role-playing is necessary or bene cial. Therefore, we opted to introduce a variable to study the e ect due to framing as well. For half of the participants, the task was framed as a study task (Framing Study), i.e., study-related user names (e.g., HXR) and passwords (e.g., HXR12345) were used. For the other half of the participants, we created a role-playing scenario (Framing Role-Play) in which they were asked to imagine they were working for a company. Thus URLs, user names, and passwords were tailored to be realistic. Naturally, such a framing variable cannot be studied within subjects, but has to be studied between subjects.

The four conditions we used in the mixed within-/between-study design can be seen in Table 1. The e ects of the di erent con guration conditions CA-Certbot and CA-Traditional were evaluated within subjects while framing e ects were evaluated between subjects.

| | | Between: Framing | |
|---|---|---|---|
| | | Role-Play | Study |
| Within: CA | CA-Cbot | 1: CA-Cbot+RP | 2: CA-Cbot+Study |
| | CA-Trad. | 3: CA-Trad.+RP | 4: CA-Trad.+Study |

Table 1: The four conditions we used in the study.

After completing each con guration task, the participants lled out an online survey that asked them about several aspects of the tasks they had performed, e.g., their self-assessment of their performance and their perception of the di culty. After completing both tasks and the questionnaires, a nal questionnaire was presented which directly compared the CA-Certbot and CA-Traditional tasks. The questionnaires can be found in appendices A and B.

## 4.2 Task Design

To tie the ndings to related work, and to allow for a better comparison, the task design was based on the study by Krombholz et al., with some modi cations as described in this section. The main task of the lab study was to acquire a certi cate for a remote Apache web server and con gure HTTPS with clear security expectations. Figure 1 shows the work ow scheme of the TLS con guration process from Krombholz et al.'s study that includes nearly all steps that are technically necessary in the manual approach, and that is similar to the CA-Traditional condition. To illustrate the Certbot automation approach, we enclosed the steps that Certbot automates with a grey box in Figure 1.

4.2.1 Sub-task 1: Baseline (SSH and Apache admin). Sub-task 1 consisted of logging on to the study server using SSH and executing some basic copy commands to place some web pages in the www directory of Apache. Sub-task 1 was used as a non-security baseline to see if participants had basic Linux skills. If participants failed in this task, their performance on the other tasks had to be taken in the context of their low Linux skill level. These two steps will be referred to as *SSH* and *Apache*.

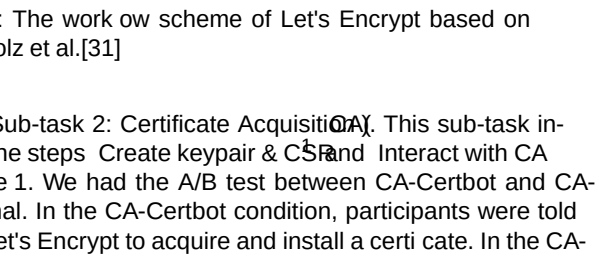


Figure 1: The work ow scheme of Let's Encrypt based on Krombholz et al.[31]

4.2.2 Sub-task 2: Certificate Acquisition (CA). This sub-task included the steps Create keypair & CSR[1] and Interact with CA of Figure 1. We had the A/B test between CA-Certbot and CA-Traditional. In the CA-Certbot condition, participants were told to use Let's Encrypt to acquire and install a certi cate. In the CA-Traditional condition, participants used a traditional CA to acquire a certi cate. Krombholz et al. used a custom minimalistic CA which did not resemble the user experience of a real CA. To make the traditional CA condition (CA-T condition) more realistic, we provided a forked version of gethttpsforfree[2]. This website resembles the steps a website administrator has to take for several o cial CAs, such as Comodo[3], and provides a guideline.

4.2.3 Sub-task 3: Configuration (Conf). In this sub-task, we had the A/B test between CA-Certbot and CA-Traditional, insofar as in the CA-Certbot condition acquisition and installation could be combined, and in the CA-Traditional condition, the participant had to manually install the certi cate acquired in sub-task 2. This task resembled the Integrate cert in Apache phase in Figure 1.

4.2.4 Sub-task 4: Configuration test. The study by Krombholz et al. ended after sub-task 3 and evaluated what participants submitted, based on criteria not known to the participants in advance. As stated, Naiakshina et al. found that students did not implement any security in a study setup unless speci ed to do so. Therefore, we speci ed the security requirements in the task description and added an explicit sub-task in which participants were asked to check their con guration using the Qualys SSL Server Test[4] tool Krombholz et al. used to evaluate the results for those participants. The details are presented in Appendix E.

4.2.5 Timeframe. Due to the within-subjects design, each participant completed the con guration task twice, once with each approach. To avoid the study seeming tedious and fatiguing, we

[1] Certi cate signing request
[2] https://gethttpsforfree.com, Accessed: 02/06/2019
[3] https://secure.instantssl.com/products/SSLIdASignup1a, Accessed: 02/06/2019
[4] https://www.ssllabs.com/ssltest/ Accessed: 09/02/2019

wanted to keep it as short as possible, while at the same time allowing enough time that participants could realistically complete the tasks. To determine the time needed, we conducted several pre-studies, and settled on a maximum editing time of three hours for the CA-Traditional task and a maximum of two hours for the CA-Certbot task. After the time limit was exceeded, the participant was asked to continue with the next condition. The observations from the pre-study suggested that if participants had not solved the tasks within these time limits, they would not be able to complete the task within the study context. Thus, we counted the participants as failing that task without making excessive demands on their time.[5]

## 4.3 Participants

One particular challenge for conducting studies with experts is acquiring a satisfactory number of participants. Therefore, we conducted the study with computer science students, because recruiting enough professional administrators for a five-hour lab study was not feasible at this stage. There is also a growing body of evidence that computer science students can serve as proxies for administrators and developers in user studies [31, 48]. In particular, Naiakshina et al. found that students are viable proxies in a password storage study in which the authors compared students to freelance developers [35]. Thus, although computer science students are not exactly the same type of user as professional administrators, we believe that they are acceptable proxies for the A/B study we conducted.

## 4.4 Recruitment and Demographics

For the first pre-study, we recruited three participants known to our group who had experience in usability studies. These participants gave feedback on the early study design.

We then recruited participants using a survey distributed via the computer science mailing list of our university. The survey was based on Krombholz et al.'s work [31] (see Appendix C). Sixty-eight participants filled out the questionnaire. Ten participants who did not fill out the questionnaire completely were removed from the selection process. We invited all 58 remaining students to participate in the lab study. Forty-five participants responded to the invitation, and 38 actually took part. Krombholz et al. found that previous experience in configuring web servers is a predictor of success. To avoid this becoming a confound, in particular because we did not exclude students with less experience, we ranked the participants based on two criteria: 1) whether they had previously configured a web server and 2) the number of correct answers in a pre-screening questionnaire. This ranking was used to build pairs of students with similar experience who were then randomly assigned to one of the two framing conditions, *Framing Study* and *Framing Role-Play*. Assignment to the CA conditions was alternated.

We conducted a second pre-study with four participants (one in each condition) to further test and improve the experimental design. This left us with 34 participants who completed the main study.

[5] In retrospect, it would have been better to give CA-Certbot the time as CA-Traditional even though it was not necessary for CA-Certbot itself. We discuss this point in the limitations sections (6).

Three participants were removed from the data set: One participant completed the first task (CA-Traditional) twice instead of each task once, and one participant successfully completed the first task (CA-Certbot) but left the study without attempting to complete the CA-Traditional task. Another participant encountered technical problems due to a temporary bug in the Certbot repository. Table 2 shows the demographics of the remaining 31 participants.

All participants were compensated with 80 Euros. We received IRB approval for the study. All participants consented to the study and signed a written consent form.

| Demographic | Number | Percent |
|---|---|---|
| Gender | | |
| Female | 3 | 10% |
| Male | 28 | 90% |
| Age | | |
| Min. | 18 | |
| Max. | 34 | |
| Median | 25 | |
| Experience as sysadmin | | |
| Yes | 22 | 71% |
| No | 7 | 22% |
| No answer | 2 | 7% |
| Configured TLS before | | |
| Yes | 15 | 48% |
| No | 16 | 52% |
| Currently employed as an administrator | | |
| Company web server | 3 | |
| Private web server | 1 | |
| Non-profit organization web server | 9 | |

Table 2: Participants' demographics ( N = 31)

## 4.5 Support Channel

The main goal of the study was to compare the usability of the CA conditions (CA-Traditional and CA-Certbot), and identify common pitfalls and potential areas of improvement. Several issues complicated this goal. First, it was important to distinguish between usability problems of the CA system and the general technical difficulties that participants might encounter. Related user studies with complex tasks showed that there is the risk of a participant failing early on, and thus, never getting to the tasks of interest [47]. Second, in relatively long procedures, such as in this study, simply asking participants to report problems at the end of the experiment runs the risk of participants forgetting some of the problems they had. It is especially likely that big problems mask smaller problems when participants recall the problems after the task.

To counter this issue, we introduced an in-scenario support channel, similar to the study pilot used by Garfinkel et al. to interact with participants [22]. We used the Mattermost chat client[6], an open source web chat platform, and a playbook (see Appendix D) to implement the support channel. Mattermost was pre-installed on all machines, and participants were told that they could message two contacts listed under direct messages named *support* and

[6] https://about.mattermost.com/ Accessed: 02/06/2019

supervisor if they encountered any problems that they could not solve on their own.

This support channel offered several benefits. First, if participants had non-CA-related difficulties, e.g., while using SSH to connect to the server, or setting permissions for copy operations, we were able to provide assistance, so that the participants were able to proceed with their main task. The fact that assistance was requested was noted, and was included in the evaluation. Second, we received feedback at the moment when problems occurred. Similar information could have been acquired using the think-aloud method, but we opted for the in-scenario channel to avoid the well-known awkwardness of the think-aloud protocol. In addition, there have been reports that think-aloud does not work well in long developer studies [36].

To ensure that the support channel would not be used inconsistently, the experimenter had to strictly adhere to the following procedure.

(1) If the question could be answered by referring the participant to the task description, this was done.
(2) If the question was a general technical question, and equally applicable to both CA conditions, help was given, and a note was made.
(3) If the question was directly related to a CA aspect of the task, the experimenter remotely analyzed what participants had done up to that point and then made the following judgment call: If the experimenter had the impression that the participant had not tried hard enough or was close to finding a solution without further help, the experimenter would respond to the participant about 10 minutes after their message to help. In addition to the couple of minutes needed to check the participant's actions, this delay was designed to raise the threshold for participants to use the support channel.[7]
If this kind of support was given, the following levels were used:
   (a) If possible, only a nudge was given. This nudge would not solve the problem but point the participant in the right direction to solve the problem without further help.
   (b) If that was unfeasible, a hint was given that would solve the specific problem; e.g., the experimenter pasted the required command in the chat, similar to how normal support staff operate.
   (c) And if that was unfeasible, the experimenter completed a sub-task for the participant, e.g., sending the CSR, sending the signed certificate, or installing the Certbot.

The last two options were last resorts. These sub-tasks were then marked as failures for the participants because they received CA-specific support. All other encounters fell into the category non-CA-specific support. Both categories are defined in more detail in section 5.4.

[7] This option turned out to not be needed, and we never had to wait 10 minutes. There was one support request which the participant solved without help even before we could have answered. In all other cases, there was ample evidence that participants had tried to solve the problem on their own first.

### 4.6 Technical Setup

The study was conducted in our usability lab which can hold up to eight participants at the same time. Each participant had a workspace with a computer running an installation of the study OS based on Ubuntu. Each participant had a set of over-ear noise canceling headphones. We also provided an overview sheet with credentials for Mattermost and Ubuntu, and a text describing the structure of the study. An example can be seen in AppendixE. The Ubuntu desktop was empty except for a link to the Mattermost chat client. The web server to be configured was running on an Amazon AWS server reachable via the domain given in the task description. Apache2 was already installed with the default configuration.

No special restrictions were introduced for the handling of the computer or the external server running the web server. The participants were equipped with root access on the server. After the task was completed, the image of the computer was automatically saved, along with the browser history, the bash history, and the Apache configuration files. Screen capture software recorded the entire procedure for the task.

### 4.7 Ethics

All participants signed a consent form with a description of the tasks and information about data collection. They were informed about the screen-recording software and the collection of their browser and bash histories. Participants were also told that we would not rate any of their solutions, and that we were interested only in the process of how they executed their tasks, to prevent an exam-like situation, which could make them feel uncomfortable or under pressure, and introduce some kind of desirability bias. The consent form, as well as the study, was approved by our university's IRB. All collected data was processed and stored in compliance with the strict general data protection regulation (GDPR) of the European Union.

## 5 RESULTS

In this section, we present the results from the lab study. We conducted qualitative and quantitative analyses. Qualitative data was collected from analyzing the discussions of the communication channel, as well as free text answers in the survey data (answered after each condition). Quantitative data was gathered from the analysis of the screen recordings of each participant in combination with the collected bash log files and the Apache2 configuration files. Unless stated otherwise, analyses were performed on the 31 participants who were exposed to both CA conditions. We found no significant differences concerning the framing variable. Therefore, the following analysis focuses on the CA variable.

### 5.1 Task Completion

In the following, we present the study participants' success rates, as well as the reasons for failure, as can be seen in Table 3. Please note that it is possible for a participant to fail at a single task and still continue on, so each column represents the local view of that step. All 31 (100%) participants succeeded in the SSH task in both conditions. Twenty-eight (90%) successfully deployed the website documents in the CA-Certbot task and 29 (94%) in the CA-Traditional task.

| | CA-Certbot | | | | CA-Traditional | | | |
|---|---|---|---|---|---|---|---|---|
| | SSH | Apache | CA | Conf | SSH | Apache | CA | Conf |
| Help+Hint | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 1 |
| Failed at | 0 | 3 | 1 | 0 | 0 | 2 | 2 | 9 |
| Not started | 0 | 0 | 2 | 3 | 0 | 0 | 2 | 5 |
| Success | 31 | 28 | 28 | 28 | 31 | 29 | 23 | 16 |

Table 3: Success rates of the participants (n = 31) divided by the CA used and the sub-tasks: 1a) SSH connection to the web server, 1b) Con guring Apache, 2) Acquiring a certi cate from the CA and 3) Con guring the web server to serve the certi cate . Help+Hint denotes the number of participants who needed CA-speci c support in the corresponding task, Failed at the number of participants who failed at the step. The participants who needed help to complete a task were counted as fails but could continue on to the next task. Some chose to give up, and are listed as Not started .

These were the two tasks we used to judge basic Linux/server con- guration skills. Twenty-eight (90%) participants in CA-Certbot and 23 (74%) participants in CA-Traditional successfully interacted with the CA and acquired a valid certi cate. Twenty-eight (90%) participants managed to correctly deploy the certi cate with Let's Encrypt, 16 (52%) using the traditional approach.

All 28 participants in the CA-Certbot and 29 participants in the CA-Traditional condition who got to the CSR stage succeeded in creating a CSR. At that point di erent problems occured. To dive deeper into the results, Table 4 provides an overview of the certi cate-related steps and problems (occurrences denoted by numbers in braces).We divided the table into four sub-groups:

> Certbot In this step, the user has to install Certbot using the operating system-dependent repository and start it.
> CSR. Then, the user has to create a key pair that is used to create a CSR. In this step, they have to choose the key size and the hash algorithm. They also have to decide for which domains the certi cate should be valid and create the actual CSR.
> Prove ownership. In this step, the user must prove that they are in control of the domain for which the certi cate will be issued. To do that, she must host a speci c le on the server the domain is pointing to. After the successful ownership veri cation, the certi cate is generated and provided.
> Certi cate Installation. Now, the user has to integrate the certi cate in the Apache2 web server, enable SSL, create a con g le, and enable the site. As an option, they can continue with a hardening phase.

For each of the steps, we highlighted in what areas knowledge or skill is useful for that step. We di erentiate between three areas: 1) Apache. For these steps, skills in con guring Apache are needed. 2) Operational. Knowledge about the operating system and how the system is to be used in the end is needed. In this case speci cally, it is knowing which domains are to be used. 3) Security. In these steps, users are exposed to security concepts and have to interact with security tools. Black circles indicate areas where a lack of knowledge or skill could lead to failing the step.

A surprising nding in our view is that the security or CA aspects did not seem to cause the participants trouble. Instead, the steps in which participants needed knowledge or skill to con gure Apache were di cult.

Three participants struggled with the ownership veri cation, where they needed to con gure the server to host a speci c le at a de ned URL. They could not manage to con gure this so that the CA could verify ownership. Two participants had problems deploying the certi cate on the web server due to the UNIX le and permission system that, e.g., prevented them from copying les. These problems seem to be problems with the handling of UNIX, Apache2, and bash, and are not directly security tasks. However, they are necessary for the con guration.

Another participant did not know that the SSL module of Apache2 has to be enabled to serve websites over HTTPS. One problem occurred because the participant created a new con guration le for a website but did not know that this site had to be enabled with a console command as well. Last, four participants could not manage to start Apache2 after the edit of the con guration le. In every on of these scenarios, we observed that the participants did troubleshooting, e.g., by searching the web or looking at video tutorials, but based on their statements, we conclude that they did not fully understand the process and the corresponding environment.

One case was particularly noteworthy: Participant P5 who started with the CA-Certbot condition failed the Apache task, i.e., did not manage to correctly con gure Apache to host the HTML les, but managed to correctly operate Certbot and completed the security con guration without task-related support. In the following CA-Traditional task, P5 managed to con gure Apache but then failed to properly install the certi cate. This lends further support to our nding that it is not lack of security skills or knowledge causing di culties: The common source of di culty is the Apache environment.

In total, 28 participants successfully managed to execute the main task in the CA-Certbot condition, whereas 16 did so in the CA-Traditional condition. McNemar's chi-square test ($p < 0.0015$, 95% con dence interval from 1.527 to 28.563) indicates a statistically signi cant higher completion rate in CA-Certbot (90%) than in CA-Traditional (52%). The McNemar test was used because we were operating on paired data.

As stated before, half of the participants interacted with Certbot rst, and the other half started with the traditional CA. In both cases, we saw that the success rates were slightly higher for the second condition, which could indicate a learning e ect. Overall, the CA-Certbot treatment had four failures when it came rst, and no failure when the task was completed as the second task. The CA-Traditional treatment had eight failures when it came rst, and seven when it came second. However, the di erences were not statistically signi cant (Fisher's exact test $p = 0.226$ and $p = 0.724$, respectively).

Table 5 gives a more detailed within-subjects view and shows the distribution of the outcome according to the number of web servers

| Step | Area | Info | CA-Certbot | Failed | CA-Traditional | Failed |
|---|---|---|---|---|---|---|
| | Apache2 Operational Security | | | | | |
| Certbot | | | | | | |
| Install | # # | | M | - | not necessary | - |
| Run | # # | | M | 1 | not necessary | - |
| CSR | | | | | | |
| Create key pair (public+private key) | ## | key size & algorithm | A | - | M | - |
| De ne domains | # # | | M | - | M | - |
| Create CSR with domains | ## | key size & algorithm | A | - | M | - |
| Prove ownership | | | | | | |
| Serve le at speci c location on web server | ## | | A | - | M | 3 |
| Certi cate Installation | | | | | | |
| Deploy certi cate | # | le permissions | A | - | M | 2 |
| Enable Apache2 SSL module | ## | | A | - | M | 1 |
| Create SSL con guration le | # | ciphers & protocols | A | - | M | 4 |
| Enable site | ## | | A | - | M | 1 |

Table 4: A detailed view of the steps and challenges of the CA and Con guration task. Beneath each step, the corresponding type of knowledge is mentioned that is needed to execute it. An M in the right columns indicates that this step has to be performed manually; A means that this step is automated.

| Number of web servers | Fail both | Success CA-Certbot only | Success CA-Trad. only | Success with both |
|---|---|---|---|---|
| 0 | 2 | 3 | 0 | 1 |
| 1 5 | 1 | 8 | 0 | 9 |
| 6 | 0 | 1 | 0 | 6 |
| Sum | 3 | 12 | 0 | 16 |

Table 5: Success rate depending on the number of web servers the participants had con gured previously.

| | CA-Certbot | CA-Traditional |
|---|---|---|
| TLD only | 8 | 1 |
| WWW only | 5 | 3 |
| Both | 15 | 13 |

Table 6: Distribution of the domains the participants chose to include in the certi cate separated by the CA condition. TLD only and WWW only mean that they entered only tld.com or www.tld.com as a valid domain.

participants reported to have con gured previously[8]. As shown, no participant who managed to successfully use CA-Traditional failed at using CA-Certbot (Success in CA-T only). However, 12 participants who succeeded with CA-Certbot failed in CA-Traditional (Success in CA-C only). Four of them started with the CA-Certbottask and eight with the CA-Traditionaltask. The results suggest that the higher the number of servers a participant had con gured previously, the fewer double failures occurred (no success in either condition). In the one to ve servers bin, roughly half the participants (eight of 18) managed only CA-Certbot, and half managed both (nine of 18). In the six or more servers bin, almost all (six of seven) managed both. This shows that Certbot (i.e., the CA-Certbot condition) is particularly useful for less experienced administrators.

However, there was one exception in which the CA-Traditional condition did better than the CA-Certbot task. It concerned the valid domain names a certi cate includes. Although not a technical speci cation, it is a common convention that tld.com points to the same website as www.tld.com . A problem that can arise is that a certi cate which is issued for only one of these domains triggers a warning for the other domain. Table 6 shows the domains that the participants chose for their certi cate. In the CA-Traditional condition, 13 participants con gured their certi cates to work for both options. Only four picked only one or the other. In the CA-Certbot condition, 15 con gured their certi cates to be valid for both options, but 13 picked only one or the other. However, this di erence was not statistically signi cant (McNemar test, $p = 1.00$).

## 5.2 E ciency

For the 16 participants who succeeded at both tasks, we observed the amount of time these participants needed to enable TLS on their server. The time was derived from the video analysis in combination with timestamps collected from the bash histories. We consider the time span as the interval from certi cate acquisition to the end of the TLS deployment process. For the CA-Certbot task, we observed a minimum time of six minutes. The maximum was 52 minutes, with a median of 18 minutes ($Mean = 21$, $SD = 15$). For the CA-Traditional task, the participants needed at least 23 minutes, and up to 113 minutes with a median of 55 ($Mean = 57$, $SD = 27$). A

[8] The questionnaire provided the answer bins 0, 1, 2 5, 6 15 and 16+. The bins 1 and 16+ had very few respondents; thus, we combined the bins with the adjacent bins for ease of analysis.

comparison of the two groups, the time participants needed for the CA-Certbot task (Median= 18) was statistically signi cantly less than for the CA-Traditional task (Median= 65; Wilcoxon signed rank test, $V = 2, p < .0027$).

## 5.3 Security Analysis

| | | CA-Cbot | CA-Trad. |
|---|---|---|---|
| Grade | A+ | 2 | 2 |
| | A | 11 | 11 |
| | A- | 0 | 3 |
| | B-F | 0 | 0 |
| | T | 3 | 0 |
| Key Size | 2048 | 15 | 0 |
| | 4096 | 0 | 16 |
| | EC256 | 1 | 0 |
| Forward Secrecy | Fully | 16 | 13 |
| | Incomplete | 0 | 1 |
| | Not Available | 0 | 2 |
| HSTS | Yes | 3 | 3 |
| | No | 13 | 13 |

Table 7: The security results we observed for each CA for participants who nished both tasks ( n = 16).

After the study was nished, we analyzed all nal server con- gurations using the Qualys SSL Server Test to identify the TLS- con guration properties, and thus, the resulting security. Qualys presents its user a rating for the server depending on the quality of their SSL con guration. Table 7 shows the outcome for the 16 participants who nished both tasks divided into the CA-Certbot group and the CA-Traditional group. Regarding the grade, nearly all con gurations got at least an A, meaning no known attacks on the protocol were exploitable, and the key size was large enough. In the CA-Certbot group, we observed three domain-name mismatches: The domain from the certi cate delivered by the server did not match the domain name from the server because the participants forgot to include www as a pre x for the domain name. This resulted in a capped grade T (not Trusted), which otherwise would have been an A-rated con guration. The reason that CA-Certbot did worse than CA-Traditional in these cases can be traced to the documentation used. In the three failure cases, the CA-Certbot participants simply followed the instruction of the tool, which does not mention or o er the www sub-domain. Whereas the tutorials used by the CA-Traditional participants made them aware of the www sub-domain, because it was suggested in an example together with the plain domain. The participants with an A+ grade extended the automatic con guration (CA-Certbot) or the manual con guration (CA-Traditional) with additional features, such as enabling HSTS. Due to the instructions given on the CA-Traditional homepage, all participants generated a key with a key size of 4096 bits compared to the 2048-bit keys generated by Certbot that were used 15 times. One participant, however, followed instructions on some website that generated the key using elliptic curves and a key size of 256 bits. Forward Secrecy was fully enabled by all 16 participants in the CA-Certbot group and 13 in the CA-Traditional group. Only one participant enabled Forward Secrecy incompletely, and two did not manage to enable it. In both conditions, three participants enabled HSTS, while all others did not.

Comparing the results to those of Krombholz et al. [31], the participants achieved higher grades. Although most of the participants' con gurations resulted in the grade B (16 of 28), and only four got an A, the participants in the present study who nished CA-Traditional ($n = 16$) were graded with at least an A- (see Table 7). However, the CA we used provided examples which the minimal CA of Krombholz et al. did not. However, only four participants had an invalid con guration in Krombholz et al.'s study, compared with 15 in the present study. This result can be explained by two factors, rstly the study set-up contained the entire process, and thus, was more complex than the Krombholz et al's study. Second, unlike Krombholz et al., we did not lter based on skill, and therefore, had a wider range of skill sets in the participant sample.

Comparing the results to Bernhard et al. [9] the participants had more success. In Bernhard et al.'s rst study zero out of nine participants managed to use the traditional CA, and only four out of nine managed with Let's Encrypt. In their second study, three out of ve managed with the traditional CA, and the same number managed with Let's Encrypt. As no details were reported at which steps the participants failed, and skill was not measured with a questionnaire but self-reported, a more detailed comparison is not possible.

## 5.4 Support

To observe the usage of the Mattermost support and feedback channel, we recorded the time and the reason for which a participant contacted us. Twenty- ve participants used the channel and asked 52 questions. Because the categorization of these messages was critical for all other results, we followed a two-stage coding procedure: First, three coders independently coded all support interactions using the categories. We calculated an initial Fleiss' kappa (0.5) and Krippendor 's alpha (0.5) [29]. With three coders and eight categories, values in this range are to be expected. All codes with disagreement were discussed, and full agreement was reached in the second round of coding. For coding categories, see Table 8.

To simplify the analysis with respect to success, we grouped participants in categories from 0 to 4 as participants who received only non-CA-speci c support. They did not receive any information relevant to the success or failure of the CA conditions that they did not already have in the task description. Category 5 participants were labeled as having received technical help while also being counted as receiving non-CA-speci c support. The distinguishing factor for technical help was that the problem had to be the same for both CA conditions, e.g., SSH or permission problems. As a counter-example, we had two participants who had problems installing Python. This was not categorized as a general technical problem, because installing Python was needed only for the CA-Certbot condition and not in the CA-Traditional condition, and thus, critical to the CA aspect. Categories 6 8 were given if questions were speci c to one of the two CA conditions. Thus, participants who needed this kind of help fell into the CA-speci c support category. Only one participant received only a single nudge; thus, category 6 did not carry much relevance for further analysis. As stated in

| | Category | Name | Description |
|---|---|---|---|
| Non-CA-specific support | 0 | No Support Contact | |
| | 1 | Self-Help | Participant solved the problem before the support experimenter had to intervene. |
| | 2 | Study Description | Questions related to information that had been handed out in the study description. The support experimenter simply repeated information from the task description. |
| | 3 | A* | Questions that went above and beyond what was expected of participants; e.g., participant asked whether we would prefer ECC over the default RSA. The support experimenter would give the answer closest to the default option. |
| | 4 | O -Topic | Messages that had no relation to the task or useful information, e.g., What is my study ID? |
| | 5 | General Technical | Problems with standard Unix commands, which a ect both CA conditions equally, e.g., problems with SSHing onto the study server. |
| CA-specific support | 6 | Nudge | Conversations where the support experimenter nudged the participants to think for themselves, e.g., answering a question by saying, This is up to you. |
| | 7 | Hint | The support experimenter sent a concrete hint for how to solve a CA-related problem, for instance, a command to run the Certbot. |
| | 8 | Active Help | The support experimenter executed part of the task for the participant, e.g., generating the signed certi cate because the participant was not likely to succeed within the time allotted, and we wanted to gather information on how the next steps would play out. |

Table 8: Support Categories

| Name | CA-Certbot | | | CA-Traditional | | | Total |
|---|---|---|---|---|---|---|---|
| | Questions | Participants | Success rate | Questions | Participants | Success rate | Questions |
| General Technical | 6 | 4 | 50% | 23 | 10 | 37% | 29 |
| Study Description | 3 | 3 | 100% | 7 | 7 | 71% | 10 |
| Active Help | 1 | 1 | 0% | 5 | 4 | 0% | 6 |
| Hint | 0 | 0 | -% | 4 | 2 | 0% | 4 |
| A* | 0 | 0 | -% | 1 | 1 | 100% | 1 |
| Nudge | 0 | 0 | -% | 1 | 1 | 0% | 1 |
| Self-Help | 0 | 0 | -% | 1 | 1 | 100% | 1 |
| Not Contacted/ O -Topic | (3) | 24 | 92% | (1) | 16 | 56% | 0 |
| Total questions | 10 | | | 42 | | | 52 |

Table 9: Support overview and success rates

section 4, interventions that fell in categories 7 and 8 were measures of last resort, and we classi ed the associated tasks as failed, but used the data separately to judge their relative di culty. For more on this, see section 5.1.

Table 9 shows the results of the support coding in descending order. The participant count does not add up to 31, because participants can be listed in multiple categories depending on the type and number of questions that they asked (excluding o -topic questions). There were almost three times as many support requests in the CA-Traditional condition compared to the CA-Certbot condition, and there were nine times as many category 7 and 8 support interventions, which indicates that the usability of Certbot is superior. A further noteworthy indicator is that 22 of 24 (92%) participants who did not contact support managed to successfully use CA-Certbot, and only 9 of 16 (56%) successfully con gured with CA-Traditional. In ve cases, the experimenter actively supported the participants (category 7 or 8) because otherwise they would not have been able to complete the task. Two participants were not able to acquire a certi cate under the CA-Traditional condition, and thus, received instructions for the installation part of the task. One participant failed to enable Apache2's mod_ssl plugin to enable TLS, and two others did not manage to restart the Apache2 web server due to an Apache con guration error. As stated before, we did not count these participants as succeeding in that condition.

## 5.5 User Feedback

After being exposed to a condition, participants were asked to ll out a survey concerning the task they had just completed. At the end of the survey for the second task, they were additionally asked to complete a nal survey on which comparative questions were asked.

5.5.1 CA-Certbot Survey. After completing the CA-Certbot task, participants were asked if they had previously heard of Let's Encrypt, and to describe the purpose of the software in their own words. All answers were gathered and coded by two researchers. Fourteen (of 31) had already heard of Let's Encrypt. We identi ed

that most of the answers mentioned that Let's Encrypt is a certificate authority (19 participants) that issues free certificates (11) to secure communication with a web server (8).

In each task survey, we asked participants which task-related steps they considered easy and which they considered hard. Of the 31 participants who finished CA-Certbot and filled out the survey, six mentioned that it was difficult to configure the Apache2 web server. For example, P2 addressed the configuration of an automatic redirect: Adding another host to the non-SSL redirects turned out [to be] annoying, Certbot did not completely fix the configuration files on expand mode. [sic] Two participants mentioned that the large amount of documentation for Let's Encrypt was hard to understand. However, one of them stated that it was still very good (P26). Finally, participants desired more information about what Certbot does, and wished to understand what is happening in the background (P28). Concerning the easy parts of the configuration process, many participants mentioned Certbot itself (12) followed by the configuration (two) and the ease of the overall process due to Certbot (two).

5.5.2 CA-Traditional Survey. Following the CA-Traditional task, we asked the same questions. Six participants reported problems with deploying the certificate in the Apache2 configuration, and four had difficulties understanding the documentation. P26 commented Each step was not very easy to understand. There should have been more details or explanations. Concerning the tasks that were perceived as easy, seven participants mentioned the documentation because it basically was just copy pasting (P11) followed by easy key generation (three).

5.5.3 Comparative Survey. In the final survey, we asked the participants to compare the two tasks in terms of the five aspects: How Easy to use, Easy to understand, Time-consuming, Transparent, and Complex were the systems?

Figure 2: Participants' perceptions of the two tasks ($n = 16$, those who succeeded in both)

Figure 2 shows the plotted outcome of this question set for participants who completed both tasks successfully. It is based on a 7-point scale ranging from 1 (CA-Certbot was better), to 4 (they were the same), to 7 (CA-Traditional was better). In all categories except Transparent, CA-Certbot performed better than CA-Traditional. It seems that the level of automation that Certbot offers reduced the perception of transparency.

# 6 LIMITATIONS

This study has several limitations that must be considered when interpreting the results. The sample consisted of computer science students from one institution. Although there is growing evidence that computer science students are useful proxies for these kinds of studies (Krombholz et al. [31], Yakdan et al. [48], Naiakshina et al. [35]), the results should not be over-interpreted and we caution against using the absolute numbers from this study to infer how a wider administrator population would fare. In particular, the trouble some of our participants had with file permissions is unlikely to affect seasoned administrators. However, it is likely that there are also varying skill levels among real administrators, and thus, we think that the insights gathered from the mix of skill levels is useful. We are also confident that the overall results of the A/B test are useful despite this limitation.

This study was also limited by the laboratory setting. It is likely that had the participants performed these tasks in a production environment with real-world security implications, they would have behaved differently. In a real setting, the participants could also have have taken more time.

Finally, the two separate time limits could have introduced a bias, which we did not think of beforehand. Although the two- and three-hour limits were grounded in the pre-studies, we did not consider the possible interaction between the two. It is possible that outcomes were affected due to a difference in learning and fatigue between the conditions. We discuss both possibilities and contrast this setup with a study setup with a three-hour limit for each of the two tasks.

Luckily, only two participants (P9 and P15) ran into the two-hour time limit for the CA-Certbot task. Both started with the CA-Certbot task. They also both failed the CA-Traditional task. If they had had three hours instead of two for the CA-Certbot task, they might have succeeded in the CA-Certbot task, and they might have learned enough during that additional hour to then also succeed in the CA-Traditional task. To judge the likelihood of either of these options, we analyzed the bash and web history of both participants. Both spent a lot of time getting familiar with the file and permission system, as well as the Apache2 configuration files. Even though it is possible that these two participants would have succeeded in their tasks if they had had one hour more, we do not think it is likely. To put this into context, participants who succeeded in the CA-Certbot task needed a median of 18 minutes (Mean= 21, SD= 15) to finish their tasks. Those who succeeded in the CA-Traditional task needed a median of 65 minutes (Mean= 57, SD= 27). Thus, although the different cut-off times were not a good design choice, they did not seem to have a negative impact on the results.

# 7 RECOMMENDATIONS

## 7.1 Recommended Improvements for Certbot

The findings presented in section 5 clearly show that the designers of Certbot have done an excellent job in making TLS configuration easier and faster. Certbot outperforms the traditional approach in almost all areas. In particular, the automation of the Apache-related tasks proved to be beneficial to the participants. But there is still room for improvement. The biggest negative aspect we found is that participants consistently ranked Certbot's transparency lower

than the manual approach, saying things like "everything was easy, but [...] Certbot is not transparent to me. I do not know what it actually did and the whole process inside, for me it is like (a) black box" (P21) or which is a little worrying for security-related tasks in my opinion" (P28). Although Certbot offers a verbose option, none of the participants made use of it. As we saw the main benefit in automating the Apache steps, it is an interesting avenue for future work to explore whether additional manual steps would have a negative or positive impact on the overall usability, security, and perception of the system.

In addition, the use of additional security features was not obvious to participants, and is not contained in the Certbots' default workflow: "The problem is that, with Certbot you cannot use HPKP, OCSP Must-Staple or Expect-CT, because you don't get a fixed private key, and no control over the CSR" (P17). Because Certbot is the recommended command line tool for Let's Encrypt, it has to cover many use cases and different types of administrators. However, offering users more advanced security configurations per default could be beneficial for the overall security. But this path has to be trodden with care. Although some experts missed advanced settings and extended configuration possibilities, we argue that Certbot is on the right path, because it is making security usable for most users. Nevertheless, future work should look into the tradeoff between security and generalizability.

A final minor observation concerns the "www" sub-domain. As stated previously, although it is not a requirement, it is a common convention that "www.tld.com" leads to the same location as the plain domain "tld.com". Currently, Certbot expects the administrator to know about this technicality and manually specify both options. Alashwali et al.'s study [6] found, that "www" domains tend to have a stronger security than their related plain domains. In this study, we saw a similar pattern, as many participants failed to include both domains. Considering the huge scale of LE and Certbot, this can lead to an even larger number of false positive warnings than Akhawe et al. found [5]. We recommend to prompting a dialog to the user that offers the option to directly issue the certificate for both domains with an explanation why this can make sense.

## 7.2 Lessons Learned from Certbot

Most academic papers highlight usability failures when examining security solutions. We studied Certbot because the general perception was that Certbot offered good usability. The study results confirm this perception. The EFF's Certbot and Let's Encrypt offer vastly better usability, leading to significantly higher success rates in less time. Therefore, we want to take this opportunity to see whether there are lessons to be learned and applied to other application areas. In our assessment, one of the key factors of Certbot's success is its simplicity born through the good design decision of a team of experts combined with good administrator-centered engineering. Participants did not need to know much about what was going on. Certbot applied the knowledge of its experts automatically with little need for specialized knowledge, by guiding the user through the process using a dialog-like approach instead of requiring multiple commands on the command line. Looking back, it is interesting to note for how long HTTPS configuration was considered a hard problem to solve at scale. Although the concept that a small group of experts decides what is best for the community is not without risk, from a usability perspective it offers a lot of potential.

The two main components of the good usability stem from automation and safe defaults. Certbot automated seven steps while introducing only two new manual steps (see Figure 4). Certbot also uses safe defaults for most security properties. The only bigger disadvantage of Certbot was that participants felt that it lacked transparency.

The question is whether Certbot' success can be replicated in other areas. For this, we need to look at several properties of the HTTPS scenario. First, we discovered that it was mainly the automation of the Apache steps that reduced failures. Although automating the other steps saved time and improved overall usability, what would classically be seen as the "difficult" steps, i.e., where the admin has to interact with cryptographic concepts, such as key generation and signing, actually did not lead to failures. As we discuss below this suggests that a good portion of research in the field of usable security and privacy might have focused on the less important parts. Second, the other implication from the fact that the Apache automation is that Certbot profits from the fact that it needs to support only a limited number of web servers. Third, there are clear recommendations about what are considered safe defaults, e.g., what key size is sufficient, what ciphers and protocol versions should be used, etc.

The attributes listed above do not lend themselves to all areas. Thus we present both scenarios in which we believe the Certbot approach can work, as well as some where other concepts need to be found.

### 7.2.1 eMail/Messaging.

Secure email is one of the bogeymen of computer security that has been plaguing usable security researchers in the end-user realm for decades [24, 40, 47]. Although standards like PGP[9] and S/MIME[10] have been around for a long time, adoption is minimal. Potentially, one of the problems is that usable security researchers have mainly targeted end-users, and not developers and administrators. Offering a simple Let's Encrypt-like service which allows administrators of an organization to roll out free and easy-to-use certificates to users, and take the burden of publishing and finding keys from them, might turn out to be a missing link. This scenario, of course, is a much more challenging than the one Let's Encrypt currently addresses. The heterogeneous environment and the large number of different components involved increase the difficulty.

Whatsapp[11], for example, hides the whole key exchange process from its users while enabling full end-to-end-encryption. Like other centralized messaging services, the engineering needed to do this is far less than in the heterogeneous email environment. However, the high adoption rate shows the promise of automating key management for end-user messaging. Thus, taking a Certbot approach to email encryption could be worthwhile, and we would like to see the usable security community look at the administrator and developer side of this old problem.

[9] https://tools.ietf.org/html/rfc3156, Accessed: 09/02/2019
[10] https://tools.ietf.org/html/rfc1847, Accessed: 09/02/2019
[11] https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf, Accessed: 09/02/2019

Password Storage. Research by Naiakshina et al. [35, 36] showed that students and software development freelancers have many difficulties when trying to store passwords securely. We see several parallels to the TLS configuration scenario. In both cases, a small number of cryptographic steps need to be taken. From the point of view of security experts, these steps are fairly easy and as HTTPS, recommended safe choices are available. But many participants did not know all steps (salting, hashing and iterations), or were not up to date. For instance, many thought that MD5 was still acceptable, or used Base64 encoding to store the passwords securely. Although many libraries offer secure storage, there is no highly visible authority and no generic approach. An initiative with a tool that can generate secure password storage code by providing a standardized and secure method by default in a number of different languages could offer similar improvements as Certbot. However, the challenge is that the number of different languages is large, and environments are more heterogeneous, which increases the technical complexity of the tool.

7.2.2 Firewall Configuration. An area of usable security research where the Certbot approach is less likely to work as well is enterprise firewall configuration. The task itself is mostly procedural; however, important security decisions specific to the administrators' goals have to be made, which was identified as a challenging area for usable security research by Edwards et al. [17]. [Administrators are often confronted with difficult decisions concerning edge cases about which packets should be discarded. Those configurations are bound to functional consequences, and giving a one fits all solution is hard. The functional steps, i.e. the configuration, can be supported with good usability [45]. However, the decisions that operators have to make cannot be easily automated, and other forms of usability research are needed.

7.2.3 Update Management. Similar to the task of firewall configuration is the case of update management for administrators who manage heterogeneous environments. Each different platform and software increases the complexity of the task and hinders simple automation. The process involves multiple stakeholders, and the decisions have consequences that impact the security and availability of systems. Previous work showed that automatic updates are not universally suitable for a corporate context [3]. The update process spans multiple stages, different policies and things to consider, such as disruptions in the others' workflow. While updates have dependencies on other parts, additional usable security research is needed.

# 8 LESSONS LEARNED CONCERNING ADMINISTRATOR STUDY DESIGN

We studied a complex administrative task in the lab to conduct an A/B comparison of Certbot and a traditional CA. As usable security research into administrators and developers is still a young field with little methodological experience, we would like to discuss insights gained from this extensive five-hour lab study.

## 8.1 Interaction via Support Channel

Allowing interaction between the experimenter and participants brings several risks. First, there is the risk that the experimenter fails to treat all participants equally. This can be countered to a certain extent by using a playbook (see Appendix D) that defines what actions an experimenter is allowed to take, and has ready-to-use texts. Second, even if the experimenter is consistent, they might still influence the results by the playbook favoring one condition or another. A careful and neutral design is needed to avoid this risk. Finally, the use of a support channel can influence the time participants need for a task and make the evaluation more complex, because the number of result categories is higher (succeeded without help, succeeded with help, failed without help, and failed with help). Despite these risks, we found the support channel offered very valuable insights into the study subject and very natural interaction. For instance, an insight we would have lost had it not been for the support channel was that one participant failed to perform the domain configuration of Apache but succeeded in using Certbot. The participant also failed to configure the traditional CA. Without the support channel, it would have looked like the participant had failed at both approaches. However, with the interaction, we saw that Certbot's usability is so good that even someone who struggles with simple configuration tasks can use it. We also gathered interesting comments and feedback from the chat. On the whole, we think the benefits outweigh the drawbacks.

## 8.2 Framing

We used two different study descriptions. One was a very simple description that made no attempt at realism or hiding the fact that it was a study task. The second introduced a role-playing scenario in an attempt to be more realistic. It used custom domains, websites, and user credentials to facilitate the role-playing scenario. We did not see any difference in behavior based on these two different frames, and thus, the substantial extra effort needed to create a more realistic study setting when designing studies for administrators in the lab context does not seem necessary. However, the lab study setup itself could have framed the participants in such a way that the scenario description did not have an influence on the outcome and that other mechanisms, e.g., field studies where the participants deploy a certificate for their own site, have to be researched. We found indicators that nudging people to security results in better security outcomes. More work is needed to analyze the influence of these factors.

## 8.3 Measuring Performance

The duration and degrees of freedom from the participants' perspective have an impact on the broad range of possible outcomes. In this study design, the participants had the possibility of choosing a non-linear way of solving the task. We used a time-consuming approach and manually tracked all user actions by watching the recorded sessions. But even then it was not easy to decide when a certain task was stopped, another one started, or a previous one was resumed. It also was hard to tell if a participant was taking a break. Requesting participants to log this would have led to an increased mental load for them, and thus, reduced the focus and created a more artificial situation. Automated approaches for this kind of task tracking would be extremely useful.

## 8.4 Expertise and Study Design

As mentioned in section 4.3, unlike Krombholz et al., we invited all students who completed the pre-screening survey to participate in the lab study, independent of their pre-screening score. The rationale for excluding low-scoring participants is to conserve study resources. There is little value in having a participant who lacks basic skills take part in an administrator study. Although it is less critical for a within-subjects design, un t participants could seriously skew between-subjects studies. However, taking only the best participants, as in the Krombholz et al. study, skews the results as well. It would be ideal to have a pre-screening survey with which to lter participants who lack the basic skills without also losing low-skilled participants. Unfortunately, our showed that most of the screening questions were not good predictors of participants' performance. In this study only the number of previously con gured servers seemed like a promising predictor.

We saw a similar picture in a developer study conducted by Wermke et al., who found a correlation between years of programming experience and success in the tasks [42]. However, a similar study by Naiakshina et al. [36] failed to nd the same correlation.

Thus, although expertise is undoubtedly important for the outcome of expert studies, assessing expertise is very hard. The di cult pre-screening process makes between-subjects study designs particularly risky, and we recommend using within-subjects designs whenever possible. At the same time, we encourage more work on assessing skill levels using questionnaires, to enable reliable balancing in future work.

# 9 CONCLUSION

We conducted a randomized control trial to compare the usability of two di erent approaches of con guring HTTPS for an Apache web server. We compared the EFF's Certbot, the recommended command line tool for Let's Encrypt CA, with a traditional approach that uses Let's Encrypt in the back-end. We showed that the EFF's Certbot is signi cantly easier and faster to use for all participants' skill levels. As a consequence of such improved usability aspects, signi cantly more users were able to set up a secure HTTPS con guration using LE than using the traditional approach. We identi ed that automation of steps pertaining to the con guration of Apache drove the increased success rate. Key generation, signing, and other cryptographic and CA-related steps did not cause the problems that might have been assumed.

Despite excellent results for Certbot, we made some minor recommendations that we hope will further improve the usability and security of LE and Certbot, and we discussed where lessons can be learned form the Certbot approach. We suggest that future work should focus on the improvement of the transparency perception of Certbot. Finally, we discussed lessons learned from conducting an administrator study and made recommendations for future studies.

# 10 ACKNOWLEDGEMENTS

This work was partially funded by the ERC Grant 678341: Frontiers of Usable Security.

# REFERENCES

[1] Yasemin Acar, Christian Stransky, Dominik Wermke, Michelle Mazurek, and Sascha Fahl. Security developer studies with github users: Exploring a convenience sample. In Symposium on Usable Privacy and Security (SOUPS), 2017.

[2] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J.Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zimmermann. Imperfect Forward Secrecy: How Di e-Hellman Fails in Practice. In Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15, pages 5 17, New York, NY, USA, 2015. ACM.

[3] Maarten Aertsen. How to bring HTTPS to the masses? Measuring issuance in the rst year of Let's Encrypt. https://www.sidnlabs.nl/downloads/theses/How-to-bring-HTTPS-to-the-masses_measuring-1y-of-LE.pdf, 2016. [Online; accessed Februar 2019].

[4] Maarten Aertsen, Maciej Korczy«ski, Giovane Moura, Samaneh Tajalizadehkhoob, and Jan van den Berg. No domain left behind: is Let's Encrypt democratizing encryption? In Proceedings of the Applied Networking Research Workshop, pages 48 54. ACM, 2017.

[5] Devdatta Akhawe, Bernhard Amann, Matthias Vallentin, and Robin Sommer. Here's my cert, so trust me, maybe? Understanding TLS errors on the web. In WWW 2013 - Proceedings of the 22nd International Conference on World Wide Web pages 59 69. International World Wide Web Conferences Steering Committee, may 2013.

[6] Eman Salem Alashwali, Pawel Szalachowski, and Andrew Martin. Does "www." mean better transport layer security? Cryptology ePrint Archive, Report 2019/941 2019. https://eprint.iacr.org/2019/941.

[7] Johanna Amann, Oliver Gasser, Quirin Scheitle, Lexi Brent, Georg Carle, and Ralph Holz. Mission accomplished?: HTTPS security after diginotar. In Proceedings of the 2017 Internet Measurement Conference, pages 325 340. ACM, 2017.

[8] Nimrod Aviram, Sebastian Schinzel, Juraj Somorovsky, Nadia Heninger, Maik Dankel, Jens Steube, Luke Valenta, David Adrian, JAlex Halderman, Viktor Dukhovni, et al. DROWN: Breaking TLS Using SSLv2. In USENIX Security Symposium, pages 689 706, 2016.

[9] Matthew Bernhard, Jonathan Sharman, Claudia Ziegler Acemyan, Philip Kortum, Dan S. Wallach, and J.Alex Halderman. On the usability of https deployment. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems CHI '19, pages 310:1 310:10, New York, NY, USA, 2019. ACM.

[10] William J. Buchanan, Scott Helme, and Alan Woodward. Analysis of the adoption of security headers in HTTP. IET Information Security, 2017.

[11] Jeremy Clark and Paul C.van Oorschot. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certi cate trust model enhancements. In Security and Privacy (SP), 2013 IEEE Symposium on, pages 511 525. IEEE, 2013.

[12] T.Dierks and E.Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), 2008. Updated by RFCs 5746, 5878, 6176.

[13] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey, and J.Alex Halderman. A Search Engine Backed by Internet-Wide Scanning. In Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15, pages 542 553, New York, NY, USA, 2015. ACM.

[14] Zakir Durumeric, James Kasten, Michael Bailey, and J.Alex Halderman. Analysis of the HTTPS Certi cate Ecosystem. In Proceedings of the 2013 Conference on Internet Measurement Conference, IMC '13, pages 291 304, New York, NY, USA, 2013. ACM.

[15] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, and J.Alex Halderman. The Matter of Heartbleed. In Proceedings of the 2014 Conference on Internet Measurement Conference, IMC '14, pages 475 488, New York, NY, USA, 2014. ACM.

[16] Zakir Durumeric, Eric Wustrow, and JAlex Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In USENIX Security Symposium, volume 8, pages 47 53, 2013.

[17] W.Keith Edwards, Erika Shehan Poole, and Jennifer Stoll. Security automation considered harmful? In Proceedings of the 2007 Workshop on New Security Paradigms NSPW '07, pages 33 42, New York, NY, USA, 2008. ACM.

[18] EFF. Certbot - About. https://certbot.e .org/about/. [Online; accessed Februar 2019].

[19] Sascha Fahl, Yasemin Acar, Henning Perl, and Matthew Smith. Why Eve and Mallory (Also) Love Webmasters: A Study on the Root Causes of SSL Miscon gurations. In Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '14, pages 507 512, New York, NY, USA, 2014. ACM.

[20] Adrienne Porter Felt, Alex Ainslie, Robert W. Reeder, Sunny Consolvo, Somas Thyagaraja, Alan Bettes, Helen Harris, and Je Grimes. Improving SSL Warnings: Comprehension and Adherence. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI '15, pages 2893 2902, New York, NY, USA, 2015. ACM.

[21] AdriennePorter Felt, RobertW. Reeder, Hazim Almuhimedi, and Sunny Consolvo. Experimenting at Scale with Google Chrome's SSL Warning. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '14, pages 2667 2670, New York, NY, USA, 2014. ACM.
[22] SimsonL. Gar nkel and RobertC. Miller. Johnny 2: A User Test of Key Continuity Management with S/MIME and Outlook Express. In Proceedings of the 2005 symposium on Usable privacy and security, pages 13 24, jan 2005.
[23] Josef Gustafsson, Gustaf Overier, Martin Arlitt, and Niklas Carlsson. A First Look at the CT Landscape: Certi cate Transparency Logs in Practice. In d Ali Kaafar, Steve Uhlig, and Johanna Amann, editors, Passive and Active Measurement, pages 87 99, Cham, 2017. Springer International Publishing.
[24] Marian Harbach, Sascha Fahl, Polina Yakovleva, and Matthew Smith. Sorry, I Don't Get It: An Analysis of Warning Message Texts. In AndrewA. Adams, Michael Brenner, and Matthew Smith, editors, Financial Cryptography and Data Security, pages 94 111, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
[25] Ralph Holz, Johanna Amann, Olivier Mehani, Matthias Wachs, and Mohamed Ali Kaafar. TLS in the wild: An Internet-wide analysis of TLS-based protocols for electronic communication. arXiv preprint arXiv:1511.00340, 2015.
[26] Ralph Holz, Yaron She er, and Peter Saint-Andre. Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS). https://tools.ietf.org/html/rfc7457, 2015. [Online; accessed Februar 2019].
[27] Saranga Komanduri, Richard Shay, PatrickGage Kelley, MichelleL. Mazurek, Lujo Bauer, Nicolas Christin, LorrieFaith Cranor, and Serge Egelman. Of passwords and people: Measuring the e ect of password-composition policies. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '11, pages 2595 2604, New York, NY, USA, 2011. ACM.
[28] Michael Kranch and Joseph Bonneau. Upgrading HTTPS in mid-air: An empirical study of strict transport security and key pinning. In NDSS, 2015.
[29] Klaus Krippendor . Reliability in content analysis: Some common misconceptions and recommendations. Human Communication Research, 30(3):411 433, jun 2004.
[30] Katharina Krombholz, Karoline Busse, Katharina Pfe er, Matthew Smith, and Emanuel von Zezschwitz. If HTTPS Were Secure, I Wouldn't Need 2FA -End User and Administrator Mental Models of HTTPS. To appear in the IEEE Symposium on Security & Privacy, May 2019, 2019.
[31] Katharina Krombholz, Wilfried Mayer, Martin Schmiedecker, and EdgarR. Weippl. "I Have No Idea What I'm Doing" - On the Usability of Deploying HTTPS. In 26th USENIX Security Symposium, USENIX Security, 2017.
[32] Let's Encrypt. Let's Encrypt Growth. https://letsencrypt.org/stats/, 2019. [Online; accessed February 2019].
[33] Frank Li, Lisa Rogers, Arunesh Mathur, Nathan Malkin, and Marshini Chetty. Keepers of the machines: Examining how system administrators manage software updates for multiple machines. In Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019), Santa Clara, CA, August 2019. USENIX Association.
[34] Antonis Manousis, Roy Ragsdale, Ben Dra n, Adwiteeya Agrawal, and Vyas Sekar. Shedding light on the adoption of let's encrypt. arXiv preprint arXiv:1611.00469, 2016.
[35] Alena Naiakshina, Anastasia Danilova, Eva Gerlitz, Emanuel von Zezschwitz, and Matthew Smith. "If you want, I can store the encrypted password." A Password-Storage Field Study with Freelance Developers. In Proceedings of the 2019 ACM SIGCHI (to appear), 2019.
[36] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, Marco Herzog, Sergej Dechand, and Matthew Smith. Why Do Developers Get Password Storage Wrong?: A Qualitative Usability Study. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS '17, pages 311 328, New York, NY, USA, 2017. ACM.
[37] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, and Matthew Smith. Deception task design in developer password studies: Exploring a student sample. In Fourteenth Symposium on Usable Privacy and Security, SOUPS 2018, Baltimore, MD, USA, August 12-14, 2018, pages 297 313, 2018.
[38] Gustaf Ouvrier, Michel Laterman, Martin Arlitt, and Niklas Carlsson. Characterizing the HTTPS trust landscape: a passive view from the edge. IEEE Communications Magazine, 55(7):36 42, 2017.
[39] RobertW. Reeder, AdriennePorter Felt, Sunny Consolvo, Nathan Malkin, Christopher Thompson, and Serge Egelman. An experience sampling study of user reactions to browser warnings in the eld. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18, pages 512:1 512:13, New York, NY, USA, 2018. ACM.
[40] Scott Ruoti, Je Andersen, Daniel Zappala, and KentE. Seamons. Why johnny still, still can't encrypt: Evaluating the usability of a modern PGP client. CoRR, abs/1510.08555, 2015.
[41] Nayanamana Samarasinghe and Mohammad Mannan. Short Paper: TLS Ecosystems in Networked Devices vs. Web Servers. In Aggelos Kiayias, editor, Financial Cryptography and Data Security, pages 533 541, Cham, 2017. Springer International Publishing.
[42] Andreas Sotirakopoulos, Kirstie Hawkey, and Konstantin Beznosov. On the Challenges in Usable Security Lab Studies: Lessons Learned from Replicating a Study on SSL Warnings. In Proceedings of the Seventh Symposium on Usable Privacy and Security, SOUPS '11, pages 3:1 3:18, New York, NY, USA, 2011. ACM.
[43] A.O. StuartSchechter, RDhamija, and IFischer. The emperor's new security indicators: An evaluation of website authentication and the e ect of role playing on usability studies. S&P, pages 51 65, 01 2007.
[44] Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and LorrieFaith Cranor. Crying Wolf: An Empirical Study of SSL Warning E ectiveness. In USENIX security symposium, pages 399 416, 2009.
[45] Benjamin VanderSloot, Johanna Amann, Matthew Bernhard, Zakir Durumeric, Michael Bailey, and J.Alex Halderman. Towards a Complete View of the Certi - cate Ecosystem. In Proceedings of the 2016 Internet Measurement Conference, IMC '16, pages 543 549, New York, NY, USA, 2016. ACM.
[46] Artem Voronkov, LeonardoHorn Iwaya, LeonardoA. Martucci, and Stefan Lindskog. Systematic literature review on usability of rewall con guration. ACM Comput. Surv., 50(6):87:1 87:35, December 2017.
[47] A.Whitten and J.D. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. Proceedings of the 8th USENIX Security Symposium, 99, 01 1999.
[48] Khaled Yakdan, Sergej Dechand, Elmar Gerhards-Padilla, and Matthew Smith. Helping Johnny to Analyze Malware: A Usability-Optimized Decompiler and Malware Analysis User Study. In 2016 IEEE Symposium on Security and Privacy (SP), pages 158 177, May 2016.

# A SURVEY AFTER BOTH TASKS

These are the questions we asked our participants after they nished each task. On a 7-point Likert scale they should rate the task di culty as well as the TLS-deployment, the certi cate acquisition and the web server con guration.

- Please enter your Study ID:
- Had you heard of Let's Encrypt before the study? (only CA-Certbot task)
- Please describe the purpose of "Let's Encrypt" in your own words: (only CA-Certbot task)
- Overall, the task was ...? (Likert)
- Which aspects were particularly di cult / easy?
- Please tell us your opinion of this task regarding the following aspects: Easy to use, Easy to understand, Time consuming, Transparent, Complicated
- Did you successfully complete the TLS con guration task? (Yes, No, Not sure)
- If you didnâ€™t nish the TLS con guration task, which steps are still missing to secure the communication?
- Overall, the process of TLS deployment was... (Likert)
- Overall, the process of acquiring a Certi cate from a CA was... (Likert)
- Which aspects were particularly di cult?
- Which aspects were particularly easy?
- Overall, the process of con guring the web server to enable HTTPS was... (Likert)
- Which aspects were particularly di cult?
- Which aspects were particularly easy?

# B FINAL SURVEY

In the nal survey we asked the participants about their security background and their experience as an administrator and how many web servers they have administered. In addition we asked them to compare the both tasks with respect to the aspects Easy to use , Easy to understand , Time consuming , Transparent and Complexity

- Please enter your Study ID:
- I have a good understanding of security concepts. (Likert: strongly disagree to strongly agree)
- How often do you ask for help when faced with security problems? (Likert: never to every time)

How often are you asked for help when somebody is facing security problems? (Likert: never to every time)
How often have you added security features to projects you were involved in? (Likert: never to every time)
Are you currently in charge of a web server? (company, private, non-pro t association,no)
Have you ever installed and con gured a web server before?
Have you ever installed and con gured SSL/TLS before?
Have you ever worked as a system administrator?
What web servers have you set up before? (e.g. * Apache, nginx,...)
How many web servers have you set up before? (0,1,2-5,6-15,> 15)
Please compare both tasks regarding the following aspects (Likert from 1 - Task 1 was better over 4 - they were the same to 7 - Task 2 was better ): Easy to use, Easy to understand, Time consuming, Transparent, Complicated
In which tasks did you enabled HSTS (HTTP Strict Transport * Security)? (Only in Task 1, Only in Task 2, In both, In none, Not sure)
Please explain your answer (Why did you enabled it? Why not? Why don't you know?).
In which tasks have you enabled HPKP (HTTP Public Key Pinning)? (Only in Task 1, Only in Task 2, In both, In none, Not sure)
Please explain your answer (Why did you enabled it? Why not? Why don't you know?).
In which tasks have you enabled OCSP-Stapling? (Only in Task 1, Only in Task 2, In both, In none, Not sure)
Please explain your answer (Why did you enabled it? Why not? Why don't you know?).
Did you use Mattermost for asking questions?
If you used Mattermost to ask questions. What was your experience of the process?
Do you think that you would have achieved the same result if you had not been able to chat with the support team via Mattermost? (yes, no)
Please explain your answer.
Thank you for answering the questions! If you have any comments or suggestions, please leave them here:

## C PRE-SCREENING QUESTIONS

This document contains the questions we asked in our pre-screening to recruit the participants. Beside some demographic information we asked them to answer bash- and web server-related questions out of which we calculated a score for each correct answer given.

Please enter your name:
Please enter your e-mail address, so we can contact you for our study:
Please enter your age:
Please enter your gender:
Which university are you at?
In which programme are you currently enrolled? (Bachelor of CS, Master of CS, other)
Your semester:
How familiar are you in using the bash-shell? (Likert: Not familiar at all to Very familiar
Have you ever con gured a web server? (yes,no)
How many years of experience do you have in programming?
How many years of experience do you have in system administration?
Which command is used to nd out the currently used IPs? (ifcon g, netstat, ipcon g, iptables,I don't know)
A symlink is created with which command? (ls -s TARGET LINK NAME, symlink TARGET LINK NAME, ln -s TARGET LINK NAME, ln TARGET LINK NAME )
TLS uses ... (symmetric cryptography, asymmetric cryptography, pem/der certi cate, X.509)
Which commands restarts the webserver? (sudo service apache2 restart, sudo /etc/init.d/apache2 restart, sudo service webserver restart, sudo service IIS restart)
Where are HTML les served by the Apache-Webserver located after default installation? (/usr/share/nginx/www, /etc/www, /var/www, /home/www)
Which is the best le permission for your private keys on a Linux system? (0777, 0300, 0644, 0600)
Please rate the security of the following Hash-functions (Likert: 1 - not secure to 7 - very secure ): Argon, MD5, BCrypt, SHA-1, RC4
Please describe the purpose of HSTS:
Certi cate Transparency is ... (providing access to the certi cates bytecode, a standard for auditing SSL certi cates, checking if a server has enabled HTTPS, a framework that helps maintaining the integrity of the SSL certi cate system)

## D ABBREVIATED MATTERMOST SUPPORT PLAYBOOK

Am I forced to use rsa keys? I could use ecdsa if I'm not bound to make use of [Own-CA-domain], as this site only permits rsa-keys. I would request the certi cate directly from LE, if you permit.
Please use the rsa keys and the Own-CA in this case.
In the survey, under Study ID shall I enter my ID, that is printed on the paper (in my case XXX), or my normal student ID?
Please enter [Study-Id].
I completed the task, the portal is available, should I con gure the apache in a special way or is the usage of the default con guration acceptable?
Since there are no other websites running I think, if it's accessible for everyone it is ne!
Must I request a new certi cate?
Yes, please do so.
I'm having a problem connecting to the server i get: Permission denied (publickey). Is it part of the task to resolve this issue?
Please use this command: `ssh -i /sshkey.pem ubuntu@DOMAINNAME.com`
I'm stuck at hosting the les. I'm trying to create a virtual host to host it

(1) Can you tell me, what have you done until now?
(2) Have you created a configuration file for apache2 in /etc/apache2/sites-available?
(3) Have you enabled the config file with a2ensite?
(4) Have you reloaded apache2?
(5) Could you please send me the contents of the .conf file?

**Is the server running or do we need to set it up?**
This is installed on the machines you connect to with ssh.
**Cannot press the 'tilde' symbol on keyboard.**
Please try ALT-Gr in combination with the "plus"-key
**Is it okay to use my email address for the use of certbot**
Please read the instructions again carefully.
**Now I am having trouble with directory as there is no such directory: home/ubuntu/website**
Please try adding a slash in front of home: /home/ubuntu/website
**How long should one wait for the result of "openssl dhparam -out dhparam.pem 4096"? With bad luck, this can take hours.**
Our experience with this command has shown that this command is executed within few minutes (< 5).
**I am trying to install apache using sudo apt-get install apache2 but it won't work.**
Please configure the apache2 instance on the server. You donâĂŹt need to install on your client.
**Is the IP for apache in browser abc.def.ghi.jkl?**
The IP for the server is [IP-Adress]
**I cannot copy from home/ubuntu/website/index.html to /var/www/html**
Please try putting sudo in front of the command.
**First I have to configure my server for url http://www.sme-company-7.com then I need to use Certificate of authority or can it be done other way?**
This is up to you. It should work both ways.
**I am trying to run ./letsencrypt-auto –apache -d www.sme-company-1.com but it is giving error**
please try these commands: "export LC_ALL="en_US.UTF-8" and "export LC_CTYPE="en_US.UTF-8" and then run it again.
**Should I use blinded@blinded.com as account email and should I generate a new key?or is a key existent**
Please use the pre-entered address and create a new key.

# E STUDYDESCRIPTION: REALISTIC SCENARIO WITH CA-CERTBOT

These are the scenario letters we handed out to participants that were in the Framing Role-Play-group and had to obtain a certificate with CA-Certbot. Page 1 contains a scenario description with additional information about the task like the command to connect to the AWS-server they had to configure. On the last page we presented them the four tasks they had to do. For each participant we modified the "URL", as well as the company name for his scenario. We blinded the descriptions for double blind review.

**STUDY HANDOUT**

TLS Configuration

Please imagine that you are the system administrator at **COMPANYNAME**, a medium-sized enterprise. Your company wants to sell their new product, the "HomeAutomator", and your boss instructed you set up the web server that hosts the product page where people can directly buy it.

Please make sure of the following:
1) Users can access **https://www.URL.com** without any warning messages
2) Make your configuration as secure as possible.

If you need any additional information to complete your task, you can contact your supervisor in your company using a chat-client called **Mattermost**:

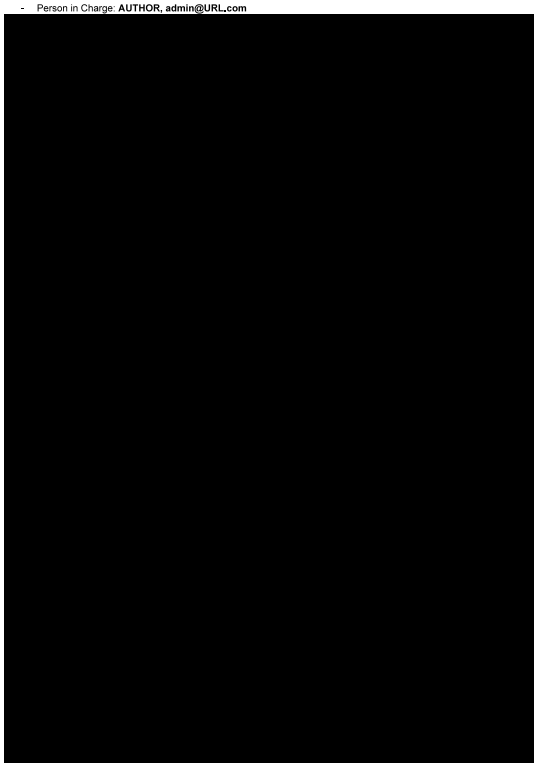
**Environment:**

You will get access to a computer with which you will connect to the server using SSH. Currently you see the Apache2 default message when you open **http://www.URL.com** on the server and an error message when opening **https://www.URL.com** .

To connect to the server, please use the following command:
- **ssh -i "~/sshkey.pem" ubuntu@URL.com**

The HTML documents for the website are already stored **on the server** and can be found at:
- **/home/ubuntu/website**

Company Details:
- **COMPANYNAME, UNIVERSITY CITY**
- Person in Charge: **AUTHOR, admin@URL.com**



# F STUDYDESCRIPTION: STUDY SCENARIO WITH CA-TRADITIONAL

This is the scenario for the Framing Study and CA-Traditional group. The structure is very similar to the realistic one except that the task is described without the company scenario.

**STUDY HANDOUT**

TLS Configuration

You're instructed to set up a web server that hosts a test website.

Please make sure of the following:

1) Users can access **https://www.URL.com** without any warning messages
2) Make your configuration as secure as possible.

If you need any additional information to complete your task, you can contact the study assistant using a chat-client called **Mattermost**:

**Environment:**

You will get access to a computer with which you will connect to the server using SSH. Currently you see the Apache2 default message when you open **http://www.URL.com** on the server and an error message when opening **https://www.URL.com**.

To connect to the server, please use the following command:

- **ssh -i "~/sshkey.pem" ubuntu@URL.com**

The HTML documents for the website are already stored **on the server** and can be found at:

- **/home/ubuntu/website**

Company Details:

- **UNIVERSITYNAME, UNIVERSITYCITY**
- Person in Charge: **AUTHOR, AUTHOREMAIL**

**YOUR TASKS:**

**Task 1 / 4**

- Please connect to the server and host the HTML files found in **/home/ubuntu/website** on the **server**

**Task 2 / 4**

- Configure the web server in a way that you can access the site with the following URLs:
  - **http://www.URL.com**
  - **https://www.URL.com**
- For this, you will have to use a Certificate Authority. Please use the following CA available at https://www.study-ca.de to acquire a free CA signed certificate for the server.
- Please use https://www.ssllabs.com/ssltest/index.html to check your configuration

**Task 3 / 4**

- Please open **https://www.URL.com** and test if you get no error message.

**Task 4 / 4**

- Please fill out the survey: https://www.surveymonkey.de/r/tls-study-ca