Sensors and Materials, Vol. 32, No. 1 (2020) 79–95
MYU Tokyo

S & M 2088

Effective Application Distribution System for Internet-less Communication during Disasters

Manato Fujimoto,[1*] Seigi Matsumoto,[1] Edgar Marko Trono,[1]
Yutaka Arakawa,[2] and Keiichi Yasumoto[1]

[1]Graduate School of Science and Technology, Nara Institute of Science and Technology,
8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan
[2]Department of Advanced Information Technology, Kyushu University,
744 Motooka, Nishi-ku, Fukuoka, Fukuoka 819-0395, Japan

(Received September 11, 2019; accepted October 28, 2019)

Keywords: Internet-less file sharing, recursive application distribution, smartphone, disaster situation

Smartphone-based disaster communication systems have been the topic of many recent studies. However, existing systems unrealistically assume that prior to a disaster, the required applications have been installed in the smartphones of all users. In disaster areas without Internet access, obtaining the software required for communications systems is difficult because application distribution platforms, such as Google Play, are web-based. In this paper, we present RecurShare, which is a novel application distribution system. The key idea of RecurShare is to be able to distribute applications without the Internet by running a web server that can be accessed using the default web browser and the tethering function of Android smartphones. We evaluated the feasibility of RecurShare experimentally and showed that RecurShare functions without problems on a variety of commercial devices. Also, user feedback suggests that RecurShare sufficiently guides users through the application distribution process in a manner that is easy to understand. Furthermore, we found that RecurShare can distribute itself at an exponential rate and that it can transmit files at speeds that are comparable to those of commonly used wireless communication systems.

1. Introduction

In recent years, many people worldwide have been accessing the Internet regularly via mobile terminals because of the ubiquity of high-functionality mobile terminals and the improvement of high-speed mobile phone lines. Even during a disaster or an accident, people can easily access useful information about the event through news sites and social networking services, or rapidly check the safety of family and friends using e-mail, messenger applications (e.g., LINE and WhatsAPP), SNS (e.g., Twitter and Facebook), and the like.

However, if a large-scale disaster disrupts Internet connections or destroys communication infrastructures, these applications and services become unavailable. Consequently, people

*Corresponding author: e-mail: manato@is.naist.jp
https://doi.org/10.18494/SAM.2020.2614

ISSN 0914-4935
https://myukk.org/

in disaster areas are rendered unable to exchange information. During such situations, systems that can efficiently exchange and share information without the Internet become very important. In particular, because people involved in disasters can only get information about their immediate surroundings, a platform that can exchange and share information is expected to markedly mitigate the feeling of insecurity of people.

Recent studies have proposed delay- or disruption-tolerant networks (DTNs) as viable alternative communication infrastructures in scenarios without Internet access.[1–7] IBR-DTN has already released a DTN implementation for the Android platform.[8] Relay-by-Smartphone, which was developed by Tohoku University, is a smartphone-based DTN that enables multihop communication without the Internet.[9] FireChat is an application available for iOS and Android that gained popularity during the large protests in Hong Kong in 2014.[10] FireChat enables "off-the-grid" communication by creating a mesh network via Bluetooth when the Internet is unavailable.

The aforementioned systems are useful and easy to distribute since they are smartphone applications. However, such systems essentially rely on the Internet because smartphone applications are generally distributed through online platforms such as Google Play and Apple Store. Thus, downloading them will be difficult during disasters when Internet connections become unavailable. Furthermore, not all users can be expected to preemptively install communication applications before disasters occur.

To address this problem, Hossmann *et al.*[11] have proposed embedding a "disaster mode" into commercial applications. They implemented a Twitter client application with a disaster mode. Their application has an emergency messaging board and enables ad hoc communication in situations where the Internet is unavailable. Ultimately, the problem may be solved if all operating systems include a disaster mode by default. However, we do not expect such a scenario to be realized in the near future.

In this paper, we propose a novel application distribution system, called "RecurShare," which solves the inconsistency of Internet-less communication applications relying on web-based distribution platforms. The key idea of RecurShare is to be able to distribute applications without the Internet by running a web server that can be accessed using the default web browser and the tethering function of Android smartphones. Specifically, RecurShare assumes that at least one seed smartphone already has the application installed. This seed smartphone is placed in a publicly accessible location, such as a municipal hall. The seed smartphone then functions as a server to which other smartphones (clients) can tether. After clients obtain a copy of RecurShare, they can then function as servers themselves and distribute the application to other clients. Through this process, RecurShare can copy itself recursively to other smartphones without requiring the Internet.

To the best of our knowledge, there are no existing works on useful application distribution systems for Internet-less communication during disasters. Also, there have been various discussions based on the assumption that disaster applications have already been distributed to the smartphones of all clients in the event of a disaster, but there have been no attempts to solve the critical problem on how to distribute those applications without the Internet during disasters.[12–15] The main contributions of this paper are summarized as follows:

- First, we develop RecurShare for the Android operating system. We address the following two main issues during the design and implementation of RecurShare: 1) how .apk installation files will be obtained from the parent terminal; 2) how .apk files will be transmitted from the parent terminal to other smartphones without requiring the Internet or special applications. To address the first issue, RecurShare leverages the feature of the Android OS wherein an application's original .apk installation files are stored in the smartphone's local directory. RecurShare accesses this directory to obtain .apk files. To address the second issue, RecurShare leverages applications and functions that are available to Android smartphones by default to distribute .apk installation files without the Internet.
- Second, we experimentally evaluate the feasibility of RecurShare to clarify its usefulness. In the evaluation, we asked four participants to distribute the RecurShare application. Because we assumed a disaster scenario, we selected participants who had no prior knowledge at all about RecurShare. The participants were graduate students in their twenties who regularly use smartphones. Commercial, non-rooted smartphones were used during the experiment. As a result, we show that RecurShare can run on a variety of commercial devices without problems and that all the participants understood how to use the application and distribute it to others.
- Finally, to test the effectiveness of RecurShare, we extend our previous work[16] by evaluating the rate at which RecurShare can be distributed. As a result, we show that the application can be distributed at an exponential rate dependent on the number of simultaneous client smartphones connected to a server smartphone. Furthermore, we evaluate the effect of file size on the distribution time. We show that RecurShare can transfer files with speeds reaching approximately 40 Mbps and that it can transmit files at speeds that are comparable to those of commonly used wireless communication systems.

The rest of this paper is organized as follows. In Sect. 2, we review existing technologies and solutions that distribute smartphone applications without the Internet. Then, we present the system design and implementation of RecurShare in Sect. 3. Next, we describe our evaluations and results in Sect. 4. Finally, we conclude the paper in Sect. 5.

2. Review of Existing Technologies

In this section, we review existing technologies that can be used to distribute smartphone applications without the Internet.

2.1 Smartphone operating systems

Although many smartphone operating systems exist, approximately 96% of the market is held by two major operating systems: Android and iOS.[17] Therefore, we focus on application distribution mechanisms for these platforms.

1) iOS: iOS is the operating system for Apple mobile devices. All applications for iOS are distributed through the Internet-based official App Store. There is no other legitimate way

to commercially distribute iOS applications. Thus, if the Internet is unavailable, iOS devices cannot legitimately download or install new applications.

2) Android: Android is an open-source operating system for mobile devices, which is developed by Google. Many low-cost devices run the Android OS, which occupies approximately 84.7% of the smartphone market. Android applications are mainly distributed through Google Play. However, applications can be distributed even without this platform. Many telecom service providers, device vendors, and third-party vendors operate their own application markets. Android also allows the direct distribution of Android application package (.apk) files, which is the format of Android application installation files. Thus, users can directly share Android applications by passing .apk files to others. Because of its open nature, we focus on the Android platform in this study and discuss methods of sharing applications without the Internet.

2.2 Internet-less .apk file distribution

Figure 1 shows different methods of distributing a file between smartphones without requiring the Internet. The simplest way is to use physical storage media such as microSD cards. Other methods use the wireless network interfaces embedded in smartphones.

1) Physical storage media: This method involves using physical storage media such as SD cards or USBs to copy and exchange files. It requires devices to have an interface for reading and writing files. However, Android devices cannot access the files stored in their built-in memory and external storage by default. Devices require a file management application to be installed. Thus, this method cannot be used in our assumed scenario, because if the Internet is unavailable, users cannot download the necessary file management applications required to read and write to storage media. In addition, some popular smartphones, such as Nexus 4 and Nexus 5, do not have storage media slots.
2) Wi-Fi Direct: Wi-Fi Direct is an interface that can be used for peer-to-peer file transmission between two Android smartphones. Devices running Android 4.0 and above come with Wi-Fi Direct by default. However, Wi-Fi Direct only provides a network connection between smartphones. If a user wants to transmit a file via Wi-Fi Direct, file management applications that support Wi-Fi Direct must be installed on both terminals. However, such applications are not usually pre-installed. Therefore, similarly to physical storage media, Wi-Fi Direct cannot be used in our scenario.

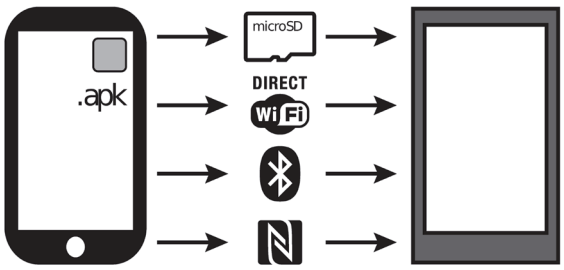

Fig. 1. Possible methods to pass .apk files without the Internet.

3) Bluetooth: Similar to Wi-Fi Direct, Bluetooth only provides a connection between two Android terminals. Therefore, it also requires file management applications that support file transmission via Bluetooth, which are not pre-installed on smartphones.
4) Near-field communication (NFC): NFC is a short-range, wireless communication technology, which is used for electronic payments. Some Android devices have an NFC technology called "Android Beam" that enables file transfer between devices in close proximity. However, the transmission speed of NFC is low and NFC only provides the identification of the application and guides users to the application's download page on the Google Play store, which requires the Internet. Thus, NFC cannot be used in our assumed scenario.

2.3 Requirements of Internet-less application distribution

To the best our knowledge, as we previously discussed, there are no suitable technologies for distributing the application without the Internet. The difficulty is that both smartphones (i.e., the sender and receiver) require specific, previously installed applications to share files. This requirement is difficult to satisfy in our assumed scenario because obtaining such applications is impossible in a disaster area where the Internet is inaccessible. Thus, a new system that can function during a disaster is required. To realize "true" Internet-less application distribution, the system must satisfy the following requirements: 1) it must be able to directly share applications between smartphones without the Internet; 2) it must not require special file-sharing applications. In the following section, we present the design of RecurShare, a novel application distribution system that satisfies the above requirements to address the inconsistency of Internet-less communication applications relying on web-based platforms for distribution.

3. RecurShare: Internet-less Application Distribution System

In the first part of this section, we present how RecurShare shares files without requiring the Internet. In the second part, we show how RecurShare recursively distributes applications.

3.1 Overview of RecurShare

As we discussed in Sect. 2, .apk files can be shared between two smartphones using a variety of wireless network interfaces such as Wi-Fi Direct, Bluetooth, and NFC. However, using these interfaces requires specific file-sharing applications to be installed in both smartphones. The assumption that all smartphones will already have file-sharing applications installed before a disaster happens is unrealistic. Therefore, the goal of RecurShare is to distribute .apk files between smartphones without requiring the Internet or special applications. We aim to realize RecurShare by utilizing only functions that are already pre-installed on Android smartphones.

To achieve this, RecurShare runs an embedded web server that distributes .apk installation files. The web server can be accessed via the web browser and tethering function that are pre-installed in Android smartphones. Clients can tether to a server terminal and use their pre-installed web browser to download .apk files. Additionally, RecurShare can guide users along the .apk-sharing process by displaying instructions on the smartphone screen.

Tethering has been implemented in Android from version 2.3. As of September 2014, 99% of Android devices are running version 2.3 or higher, thus we consider tethering to be a pre-installed function. Similarly, Android devices come with a web browser. Because RecurShare only uses pre-installed functions, it does not require specific applications to be pre-installed. The requirements of smartphones running RecurShare (i.e., the senders/servers and receivers/clients) are summarized in Table 1.

3.2 Internet-less file sharing between smartphones

Figure 2 shows the layered architecture of RecurShare. RecurShare is installed on the smartphone on the left of Fig. 2 (i.e., the sender), while the smartphone on the right of Fig. 2 (i.e., the receiver) is downloading a copy of RecurShare.

RecurShare has the following functions: a visual guide, a web server, and tethering settings. When a sender runs RecurShare, a visual guide is shown on the display. The guide shows how the user can set up the tethering function.

After specifying the settings of the tethering function, such as the access point name and password, a receiver can attempt to connect to the sender smartphone. Once successfully connected, a temporary local network is established between the sender and receiver smartphones. In a typical use case, tethering shares Internet access through a Wi-Fi network. RecurShare uses tethering only to establish a temporary local network.

After establishing a connection, the receiver uses its default web browser to access the web server embedded in RecurShare to download the application. We implemented the web server using the open source software named nanoHttpd. We programmed custom responses to http access requests from a receiver to prompt the installation directly after downloading the .apk. If the `content-type` of the received file is .apk, the web browser installs the file on the device.

Table 1
Requirements of RecurShare.

User	Wi-Fi	Web browser	Tethering
Sender (Server)	Required (default)	Not required	Required (default)
Receiver (Client)	Required (default)	Required (default)	Not required

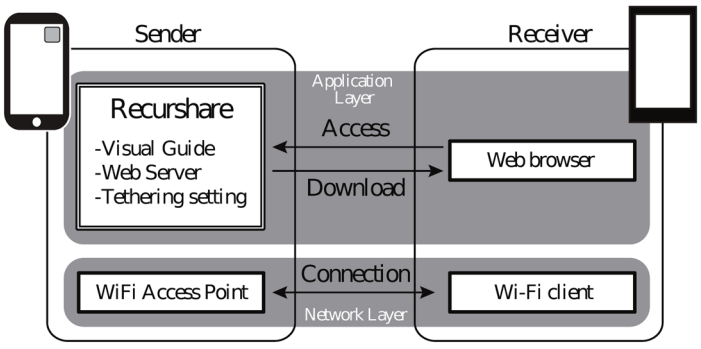

Fig. 2. RecurShare architecture (network and application layers).

3.3 Recursive application distribution

To achieve widespread and high-speed distribution, we propose recursive distribution through a self-reproduction technique. Figure 3 shows how RecurShare duplicates its own .apk file and distributes it.

Initially, a seed smartphone is assumed to already have a copy of RecurShare. The seed smartphone runs the web server (i.e., the server), and client smartphones can tether to the server smartphone and download the RecurShare .apk. After client smartphones download and install RecurShare, they can function as servers and further distribute RecurShare to other client smartphones. If multiple clients connect to the server, RecurShare can simultaneously distribute itself to many smartphones. As such, the distribution rate of RecurShare can be represented by a geometric progression.

RecurShare accesses the .apk files stored in a smartphone for recursive distribution. In the Android OS, all original .apk files are stored in a local directory after installation. These files are located in `/data/app/`.

Root permission is required to access the `/data/` directory. However, if the full path to a file is accessed, for example, `/data/app/filename.apk`, root permission is not required.

The full path of the target .apk file can be obtained from the package manager using the command `adb shell pm list packages -f | grep package_name`. We used this to obtain the original .apk of RecurShare without root permission. Once obtained, the .apk can be distributed to other devices.

3.4 Distribution flow of RecurShare

Figure 4 shows how RecurShare distributes applications. We assume the existence of a seed smartphone in which RecurShare has already been installed.

When a user runs RecurShare as a server, the application first checks the state of the smartphone tethering function. If tethering is disabled, a tethering setting button is displayed on the screen. If tethering is already enabled, the information required (i.e., SSID, download page URL, and QR code) for connecting to the device will be displayed on the screen.

Based on the displayed information (i.e., SSID and password), client smartphones will attempt to access the server device through their Wi-Fi interface. After establishing the connection, the clients use their web browsers to attempt to access a web page to download

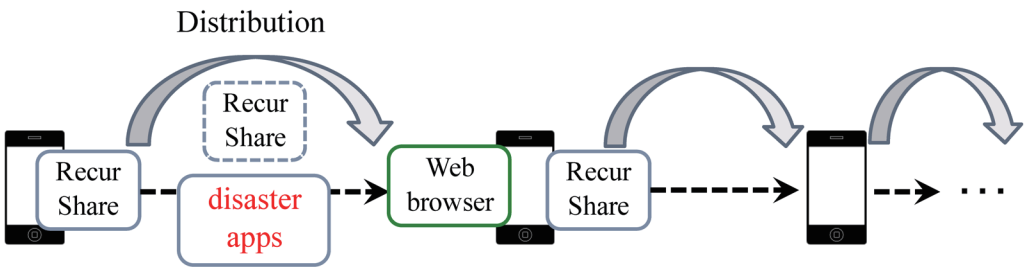

Fig. 3. (Color online) Recursive distribution.

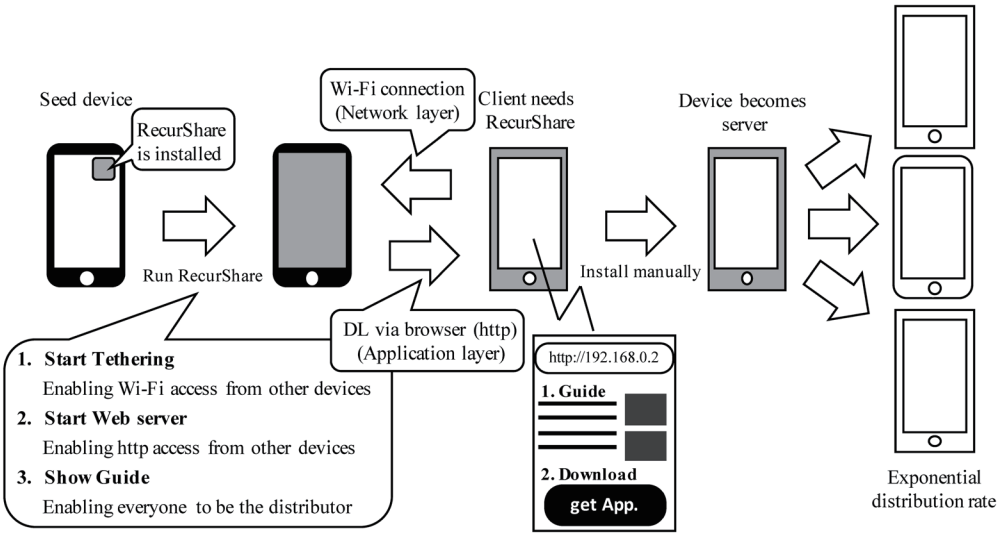

Fig. 4. Distribution flow.

applications. The web page is rendered on the embedded web server in the sender smartphone. Since there is no Domain Name System (DNS) in this tentative network, the URL of the download page is expressed through IP addresses, such as `http://192.168.0.2`. Alternatively, a QR code of the URL is displayed on the server smartphone's screen. Client smartphones can scan this QR code to access the download page. NFC can also be used to share the URL.

The download page displays instructions on how to download the .apk and how to install the application after the download completes. The user of the client smartphone follows the instructions and obtains the .apk from the server.

RecurShare can copy itself and other applications installed on the server smartphone. However, we limit the downloadable applications in the current version because of the copyright and security reasons discussed in the next subsection.

The user of the client smartphone then installs the application manually because Android devices ask for a confirmation before installing .apk files from unknown sources. Once allowed, RecurShare is installed on the client smartphone.

After RecurShare is installed and run, the client smartphone can then function as a server and distribute RecurShare to other clients. Repeating this process allows RecurShare and other applications to be distributed without the Internet.

3.5 Security

In this study, we assume that RecurShare is used only during emergencies or post-disaster scenarios. However, if RecurShare is used in other scenarios, it is entirely possible for it to be used maliciously, for example, to distribute viruses or malware. Therefore, the security issues that arise from using RecurShare must be considered. In this subsection, we present some countermeasures that are under development, which can prevent undesired usage.

1) Activation mechanism for the redistribution function: We propose a local activation mechanism for the distribution function to prevent the undesired distribution of applications. RecurShare instances installed on new smartphones (called "child RecurShare") must be turned on once when the receiver is in the same network as the sender smartphone. The child RecurShare instance sends an activation message to the RecurShare instance installed on the server smartphone (called "parent RecurShare"). The parent RecurShare instance checks the validity of the child through a cyclic redundancy check (CRC) sent from the child RecurShare. After confirming the validity, it grants permission to redistribute the obtained .apk to others in exchange for the child's MAC address.
2) Distribution routes: Using the proposed activation mechanism, all parent RecurShare instances can record the MAC address of each associated child RecurShare. All RecurShare instances upload the collected MAC address to a central server when an Internet connection becomes available. From this, the central server can calculate the distribution routes and times from a parent to its children.
3) Controlling distribution range: Limiting the distribution range is critical in minimizing damage from potentially malicious users. To limit the range, we consider the following parameters: network availability, time, location, and the number of hops.
   —Network-based limitation: We can limit RecurShare to be used only when there is no Internet connection. If an Internet connection becomes available, RecurShare is automatically stopped after uploading the collected MAC address list.
   —Time-based limitation: Two time parameters can be set to limit the distribution. If we use an absolute time such as `until 2014/11/20 23:59:59`, we can set the time when the application expires. If we use a relative time such as `2 hours from the first run`, we can manage the duration of availability of applications. Even if a smartphone's clock is changed intentionally to avoid this control mechanism, discrepancies can be detected by comparing the device times of the server and client. If there is a significant difference, RecurShare stops distribution.
   —Location-based limitation: We assume that RecurShare is used for distributing Internet-less applications within a relatively short communication range. Therefore, distribution can be controlled by specifying a maximum allowable distribution range using location parameters such as `center coordinate: x, y, range: 10 km`.
   —Hop-based limitation: The number of hops (or steps) is a promising method of control for RecurShare. Although RecurShare using the hop-based limitation can distribute applications at an exponential rate until a predetermined number of hops (number of descendants) is reached, it immediately stops distributing applications when a predetermined number of hops is reached. Specifically, by counting hops and activated child RecurShare instances, RecurShare can stop distribution once a maximum hop count is reached. Hence, the distribution rate of the hop-based limitation is the same as the unrestricted RecurShare, but it can limit the distribution range compared with the unrestricted RecurShare.
4) Server-client authentication: In the RecurShare system, if tethering is successful, anyone can access the web server easily. Therefore, it is necessary to add security measures

between servers and clients. As shown in Fig. 5, RecurShare consists of a sender (server) and receivers (clients). The receivers need to access the server smartphone of the sender directly via Wi-Fi connection. When establishing a Wi-Fi connection, the sender instructs both the AP name and password orally or by showing own smartphone's screen. Non-malicious receivers near the sender can be easily authenticated if they can view and input the displayed password. On the other hand, malicious receivers who remotely overhear the AP cannot view or input the displayed password and will not be granted access. Therefore, the receivers are limited to only non-malicious persons around the sender.

5) Controlling clients who can select applications to distribute: To prevent the spread of applications containing viruses or malwares by malicious clients, it is very important to limit clients who can select applications to distribute. In RecurShare, basically anyone can select the application to distribute, but it is necessary that clients who can select applications to distribute are limited to only clients who are authorized to access the server smartphone of the sender. RecurShare can limit clients who can access the server smartphone by using the proposed server-client authentication. Therefore, accesses from strangers and potentially malicious clients can be prevented, and the spread of applications containing viruses or malwares can be suppressed. Also, we assume that an initial sender is a person in local governments who never distributes viruses or malwares intentionally. Hence, viruses and malwares will not spread unless the initial sender intentionally distributes them.

4. Performance Evaluations

In this section, we evaluate our proposed Internet-less application distribution mechanism. We conducted three evaluations: 1) we asked four participants to evaluate the feasibility of RecurShare, 2) we calculated the distribution rate of RecurShare, and 3) we evaluated the effect of the file size on the distribution time.

4.1 Feasibility evaluation

We conducted an experiment where we asked four participants to use RecurShare to distribute an .apk file to other users. We measured the time taken by the participants to

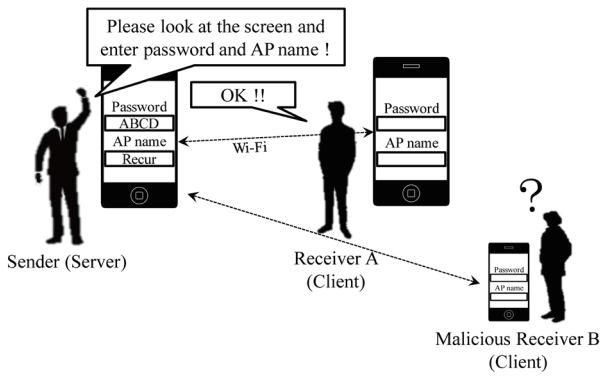

Fig. 5. Server-client authentication.

complete the .apk distribution process. At the end of the experiment, we asked the participants to answer a questionnaire about their experience.

4.1.1 Implementation of RecurShare

We implemented a simple version of RecurShare that can only distribute the RecurShare .apk. Screenshots of the RecurShare application running on the sender smartphone and a web browser running on the receiver smartphone are shown in Fig. 6.

Figure 6(a) shows the tethering settings screen on the sender smartphone. The sender sets up tethering in this screen.

Figure 6(b) shows the main screen on the RecurShare sender smartphone. After setting-up tethering, RecurShare displays this screen.

Figure 6(c) shows the download page on the receiver smartphone's browser. When the receiver accesses the sender's web server through the tethered network, this page is displayed. The receiver obtains a visual guide that describes how to download the application and how to install it. After reading the guide, the receiver starts downloading the application by tapping the download button.

Figure 6(d) shows the screenshot when the receiver successfully downloads the RecurShare .apk. Once the download is accepted and finished, RecurShare is installed on the receiver smartphone.

4.1.2 Experimental setup

The experiment was conducted inside a classroom with no wireless networks to emulate an evacuation center. We chose four participants who had no previous knowledge about RecurShare.

In the classroom, we set one seed smartphone that had the RecurShare application. The first participant used the seed smartphone to distribute RecurShare to the next participant. Once a participant had successfully installed RecurShare, the participant again distributed it according to the instructions.

(a) (b) (c) (d)

Fig. 6. (Color online) Screenshots of RecurShare: (a) set up tethering (sender), (b) main screen (sender), (c) web site shown on receiver's browser, and (d) successful download of .apk.

We measured the elapsed time from when the sender starts RecurShare to when the receiver starts RecurShare after installing it. Although RecurShare can distribute applications simultaneously to multiple tethered receivers, the sender sent the .apk to only one receiver in this experiment. The experiment is schematically shown in Fig. 7.

4.1.3 Participant questionnaire

The participants answered the following questions on a scale of 1 (worst) to 5 (best).

1. Were you able to distribute the application according to the visual guide shown on the parent RecurShare?
2. Would you want to use RecurShare during actual disasters?

In addition, we asked the participants for their feedback about the possible security risks when using RecurShare.

4.1.4 Results

All the participants correctly operated RecurShare and distributed the application on their first attempt. This shows how the visual guide helps first-time RecurShare users understand how to correctly operate the application.

The best, worst, and average elapsed times were 82, 166, and 111 s, respectively. These results show that RecurShare can distribute applications within a reasonable duration. However, to accommodate people who are not good at using smartphones, the user experience and user interface of RecurShare may need to be improved.

The answers to question (1) were as follows: three participants gave the best score of 5 (easy to understand), and one participant gave a score of 2 (the guide should be improved). For question (2), the average score was 3.5. This shows that most of the participants found the application useful and would want to use RecurShare during disasters.

However, when we asked whether they "wanted to use the application regularly," we received mixed responses. Three participants gave the lowest score of one (do not want to use it). This

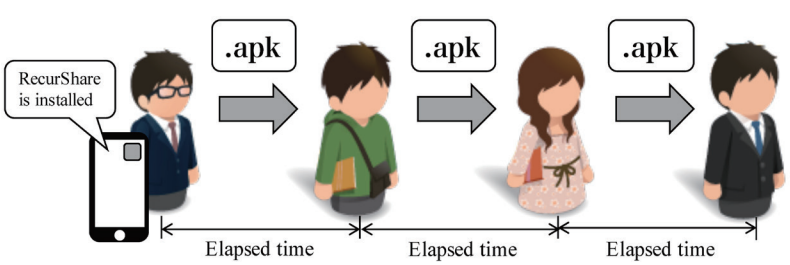

Fig. 7. (Color online) Feasibility experiment.

may signify the participants' anxiety over security issues. Nevertheless, if a city or trusted organization were to advocate an official version of the application, the anxiety may be reduced.

When asked about their thoughts on the security risks that RecurShare may pose, the participants gave the following responses:

1. RecurShare has some security risks, but it will not be a big problem because there may not be many malicious users when there is no Internet.
2. I really want to use RecurShare and get an application from unknown people, because it sounds helpful during emergencies.
3. In a disaster, the usefulness is more important than the security risks.
4. I want to use RecurShare under normal circumstances because it looks useful. However, I do not want to use RecurShare during a disaster situation with unknown people. I am afraid of security risks, although I do not know any concrete risks of RecurShare.

4.2 Calculating distribution speed from results of feasibility evaluation

From the results of the feasibility evaluation, we calculated the theoretical distribution rate of RecurShare. In this evaluation, we made the following assumptions and definitions.

1. We assume an evacuation center with 500 evacuees in a disaster area.
2. The evacuation center has no access to the Internet because the disaster has destroyed the area's communication network infrastructure.
3. One evacuee in the center has a smartphone with the RecurShare application (i.e., the seed). All other smartphones in the center do not have the RecurShare application.
4. Smartphones that already have RecurShare and send .apk files are referred to as servers.
5. Smartphones that do not have RecurShare and receive .apk files are referred to as clients.
6. For simplicity, we assume a constant .apk file transmission time from one smartphone to another.

The goal in the scenario is to distribute the RecurShare .apk to all smartphones in the evacuation center. We measured the distribution rate of RecurShare by the number of time intervals, τ. At the k-th time interval ($1 \leq k \leq \tau$), each server distributes the RecurShare .apk to a maximum of n clients. When a client receives the .apk, it installs RecurShare and becomes a server at the $(k + 1)$-th time interval. We define the number of smartphones that have installed RecurShare at the τ-th interval as

$$Nmax = \alpha^{\tau}, \tag{1}$$

where $\alpha = n + 1$. For this scenario, at $\tau = 0$, only the seed smartphone is the server. We want to obtain τ for the target $Nmax \geq 500$ (i.e., 500 smartphones in the area).

In our evaluations, we calculated τ for the following two cases:

1. $n = 1$ (i.e., each server can distribute the RecurShare .apk to 1 client per time interval, τ) and
2. $n = 7$ (i.e., each server can distribute the RecurShare .apk to 7 clients per time interval, τ).

We chose $n = 1$ because it represents the simplest operation scenario where one server shares a file with one client at a time. $n = 7$ was chosen because real-life Android devices can only tether up to seven clients at a time.

For $n = 1$, $Nmax \geq 500$ when $\tau = 9$ (i.e., $Nmax = 512$). From the results of our feasibility experiment in Sect. 4.1.4, we found that the average time to send the RecurShare .apk from a server to a client (i.e., the actual length of each time interval τ) was approximately 2 min. In that experiment, however, all clients were already in close proximity to the server. In a real-life scenario, additional time is required for clients to search for a server. To represent the search time, we increased the time interval τ to 5 min. Given this assumption, $\tau = 9$ will require 45 min.

For $n = 7$, $Nmax \geq 500$ when $\tau = 3$ (i.e., $Nmax = 512$). This shows that the distribution rate of RecurShare exponentially increases with the number of simultaneously connected clients (n). In this case, if the time interval is 5 min, $\tau = 3$ will only require 15 min.

From these calculations, we showed that RecurShare is capable of distributing .apk files at a very high speed.

4.3 Evaluating effect of file size

For our final experiment, we evaluated the impact of file size on the distribution time. To measure the distribution time, we implemented a time logging application for client smartphones. The application records the client's system clock at two points: (1) when the request message to acquire the .apk file from the server is sent and (2) when the download of the .apk file finishes. The distribution time was obtained from the difference between (1) and (2).

In our experiments, we distributed existing applications that enable Internet-less communication. These applications had different .apk file sizes and are listed in Table 2. We also distributed other applications, such as games and useful tools, to evaluate the performance for larger .apk file sizes as shown in Table 3.

We measured the distribution time of each .apk file and calculated the average of five trials. The distribution times and derived transfer rates are listed in Table 4 and graphed in Figs. 8 and 9, respectively.

Figure 8 shows that the distribution time directly correlates with the file size. The distribution time of applications with file sizes from 0 to 2 MB was within 1 s. The IBR-DTN application had the largest .apk file size (9.58 MB) among the Internet-less communication applications and required approximately 2.2 s to distribute. The distribution time of the RecurShare application obtained during the feasibility evaluation was approximately 0.5 s. For game applications with large file sizes, it took approximately 6 s for a 30 MB .apk file to be distributed and 9.2 s for a 47 MB .apk file. In the feasibility evaluations, it took approximately 2 min to distribute an application; the time delay caused by distributing .apk files with larger sizes is negligible.

Table 2
Internet-less communication applications.

Application name	File size (MB)	Outline
Bluetooth Chat	0.42	Bluetooth chat application
Whisper	0.85	Information exchange application
Open Garden	2.01	MANET-based application
RecurChat[18]	3.58	BLE-based chat application
FireChat	6.15	Chat application
IBR-DTN	9.58	DTN-based application

Table 3
Games and Internet-less tools.

Application name	File size (MB)	Outline
Galaxy Reversi	2.00	Bluetooth battle reversi
Flashlight	4.85	Flashlight application
UMZ Mini Games	29.44	Bluetooth communication game pack
Virtual Table Tennis	46.72	Bluetooth battle table tennis game

Table 4
Distribution times and transfer rates.

Application name	File size (MB)	Distribution time (s)	Rate (MB/s)
Bluetooth Chat	0.42	0.427	0.98
RecurShare	0.79	0.531	1.49
Whisper	0.85	0.493	1.72
Galaxy Reversi	2.00	0.788	2.54
Open Garden	2.01	0.673	2.99
RecurChat[18]	3.58	1.058	3.38
Flashlight	4.85	1.396	3.47
FireChat	6.15	1.545	3.98
IBR-DTN	9.58	2.230	4.30
UMZ Mini Games	29.44	5.861	5.02
Virtual Table Tennis	46.72	9.118	5.12

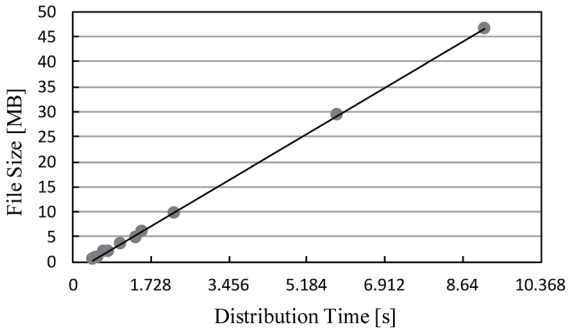

Fig. 8. Distribution time for each application size.

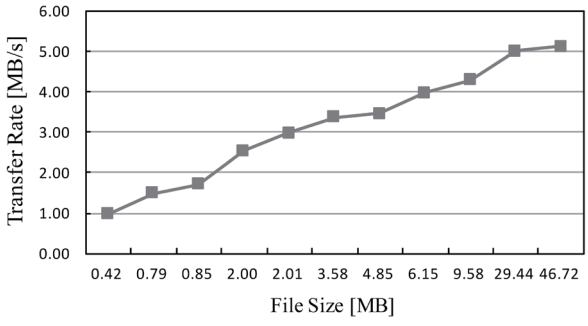

Fig. 9. Transfer rate for each application size.

Figure 9 shows the data transfer rate for each application. The curve shows that the maximum data transfer rate of RecurShare was approximately 5 MB/s (40 Mbps). This shows that the speed of RecurShare is comparable to those of commonly used wireless communication systems.

5. Conclusion

In this paper, we proposed RecurShare, a novel Internet-less application distribution system. Also, we addressed the inconsistency that Internet-less communication applications, such as FireChat, require web-based distribution platforms. During disasters, the Internet is often inaccessible, and the assumption that Internet-less communication applications have already been installed in all users' devices is unrealistic.

To realize "true" Internet-less communication, RecurShare uses the pre-installed web browser and tethering functions of Android smartphones. By embedding a web server in RecurShare and activating tethering, other smartphones can download .apk files using their web browsers. Furthermore, because RecurShare guides users through the application distribution process by displaying instructions on the smartphone screen, users can easily understand how to distribute and install applications. Finally, RecurShare does not require root permission and allows recursive distribution.

From experimental evaluations, we found that the users were able to operate RecurShare to distribute applications without the Internet within a reasonable time. Furthermore, user feedback suggested that the participants found it easy to follow the instructions. We also found that RecurShare can distribute itself at an exponential rate and that it can transmit files at speeds that are comparable to those of commonly used wireless communication systems.

In the future, we will evaluate the performance of RecurShare at a city-wide scale through computer simulations and experiments. Also, we will plan to design a user interface that can be easily operated even by users who have never used RecurShare, e.g., for the placement of large buttons and multilingual support for foreigners.

Acknowledgments

This research was partly supported by JSPS KAKENHI (JP15H05708 & JP19H01139), JST PRESTO (JP16817861), and the Cooperative Research Project of the Research Institute of Electrical Communication, Tohoku University.

References

1 K. Fall: Proc. 2003 Conf. Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM, 2003) 27. http://doi.acm.org/10.1145/863955.863960
2 S. Jain, K. Fall, and R. Patra: Proc. 2004 Conf. Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM, 2004) 145. http://doi.acm.org/10.1145/1015467.1015484
3 A. V. Vasilakos, Y. Zhang, and T. Spyropoulos: Delay Tolerant Networks: Protocols and Applications (CRC Press, Boca Raton, Florida, 2012) pp. 1–362.
4 K. Wei, X. Liang, and K. Xu: IEEE Commun. Surv. Tutorials **16** (2014) 556. https://doi.org/10.1109/SURV.2013.042313.00103
5 E. M. Trono, M. Fujimoto, H. Suwa, Y. Arakawa, and K. Yasumoto: Comput. Commun. **100** (2017) 129. https://doi.org/10.1016/j.comcom.2016.12.003
6 J. Fajardo, K. Yasumoto, N. Shibata, W. Sun, and M. Ito: J. Inf. Process. **22** (2014) 106. https://doi.org/10.2197/ipsjjip.22.106
7 L. Pelusi, A. Passarella, and M. Conti: IEEE Commun. Mag. **44** (2006) 134. https://doi.org/10.1109/MCOM.2006.248176

8 J. Morgenroth, S. Schildt, and L. Wolf: Proc. 18th Annual Int. Conf. Mobile Computing and Networking (ACM, 2012) 443. http://doi.acm.org/10.1145/2348543.2348606
9 H. Nishiyama, M. Ito, and N. Kato: IEEE Commun. Mag. **52** (2014) 56. https://doi.org/10.1109/MCOM.2014.6807947
10 Open Garden: http://opengarden.com/firechat (accessed September 2019).
11 T. Hossmann, F. Legendre, P. Carta, P. Gunningberg, and C. Rohner: Proc. 3rd Extreme Conf. Communication (ACM, 2011) 1:1. http://doi.acm.org/10.1145/2414393.2414394
12 J.-S. Lee and S.-B. Yang: Proc. Service-Oriented Computing and Applications (IEEE, 2010) 1. https://doi.org/10.1109/SOCA.2010.5707159
13 Y. Nakajima, H. Shiina, S. Yamane, T. Ishida, and H. Yamaki: Proc. Int. Symp. Applications and the Internet 2007 (IEEE, 2007) 2. https://doi.org/10.1109/SAINT.2007.13
14 K. Mori, A. Yamane, Y. Hayakawa, T. Wada, K. Ohtsuki, and H. Okada: IEICE Trans. Fundam. **E96-A** (2013) 649. https://doi.org/10.1587/transfun.E96.A.649
15 T. Wada, H. Higuchi, K. Komaki, H. Iwahashi, and K. Ohtsuki: J. Adv. Simul. Sci. Eng. **3** (2016) 79. https://doi.org/10.15748/jasse.3.79
16 S. Matsumoto, Y. Arakawa, E. M. Trono, and K. Yasumoto: Proc. 5th Int. Workshop Pervasive Networks for Emergency Management (IEEE, 2015) 512. https://doi.org/10.1109/PERCOMW.2015.7134090
17 IDC: https://www.idc.com/promo/smartphone-market-share/os (Accessed January 2018).
18 M. Fujimoto, S. Matsumoto, Y. Arakawa, and K. Yasumoto: Proc. 9th Int. Conf. Mobile Computing and Ubiquitous Networking (IEEE, 2016) 1. https://doi.org/10.1109/ICMU.2016.7742096