Secure Bootstrapping of Cloud-Managed Ubiquitous Displays

Mohit Sethi^{*†}, Elena Oat[†], Mario Di Francesco[†], Tuomas Aura[†]

{mohit.sethi, elena.oat, mario.di.francesco, tuomas.aura}@aalto.fi

^{*}NomadicLab, Ericsson Research, Finland

[†]Aalto University, Finland

ABSTRACT

Eventually, all printed signs and bulletins will be replaced by electronic displays, which are wirelessly connected to the Internet and cloud-based services. Deploying such ubiquitous displays can be cumbersome since they need to be correctly configured and authorized to access both the Internet and the necessary services, despite the fact that they have minimal input capabilities and may be in inaccessible locations. Our goal is to enable easy and secure configuration of ubiquitous displays such as digital signage and advertisements, which are managed by cloud services and show HTML5 content. In our solution, the display shows a QR code which, when scanned by the user with a camera phone, allows automatic configuration of the wireless network along with the content to be shown. This is accomplished by a long-term trust relation configured between the cloud service and the wireless access network. We build on existing technologies and standard protocols, including RADIUS and EAP, without requiring new software to be installed on the phone or changes to the network infrastructure.

Author Keywords

Displays; security; WiFi; access-point; EAP; QR code; smart phone; configuration; bootstrapping; digital signage; cloud.

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces

General Terms

Human Factors; Security; Management.

INTRODUCTION

Electronic displays have become ubiquitous in urban environments, including public, semi-public and private spaces such as shops, cafés, transport hubs, streets, offices, classrooms as well as homes. The displays fulfill varying purposes from disseminating official information to advertising and from entertainment and art to personal communication. New technologies such as digital paper [18] and wireless charging [14]

will help to push down the cost, broaden the range of display sizes, and improve their power efficiency, thus enabling new applications. Many of the new displays are replacing functions previously served by paper and ink: traffic and safety notices, time tables, advertising billboards, office signs, price tags, photo frames etc. Eventually, we could expect informal and ad-hoc communication, such as public bulletins, posters, sticky notes, to also move to a digital format. Overall, the developments in displays are part of the general trend of services first moving to the Internet and then back to the physical space, but in a new, electronic and connected form that enables both remote and local access.

We believe that most of these displays will be *connected wirelessly to the Internet* and that they will be *managed by and served content from servers in the cloud*. The management of these cloud-connected displays is profoundly different both from the current management of organizational assets, which are mostly administered locally and connected at most to local servers, and from personal devices, which are still mostly paired directly with each other. The management service and content may be personalized, but it is likely that both will be provided by open cloud services, which will be partly funded by advertising. For the display owners, it will remain to install the devices to their physical locations, to provide them with Internet connectivity through IEEE 802.11 [21] and similar wireless standards, and to choose the desired content.

In this paper, we open the investigation into the communication and security aspects of such cloud-managed ubiquitous displays. In particular, we address the issue of secure deployment and establishment of the necessary trust relations. The main challenges are the large number of displays anticipated, and the need to support the ad-hoc deployment of off-the-shelf hardware. Organizations and individuals should be able to deploy any number of displays with minimal administrative overhead and no special training. These may be new display devices just purchased or existing devices that have been reset to their default configuration and are being redeployed for a new purpose. While there has been previous research into display-related technologies ranging from display hardware [19, 20] to social impacts of public displays [32], there have not been many solutions to support their efficient installation and configuration in the research literature.

The configuration of cloud-managed displays consists of two main steps: (1) enabling Internet access through a local wireless network and (2) connecting to a given user account on a specific cloud-based management service. After that, (3) the

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author. Copyright is held by the owner/author(s).

UbiComp '14, September 13-17, 2014, Seattle, WA, USA
ACM 978-1-4503-2968-2/14/09.

<http://dx.doi.org/10.1145/2632048.2632049>

asset management and content selection can be done with a web-based user interface. It is our goal to make the two initial steps as smooth as possible, while also ensuring the security of the setup process. The task is made difficult because the displays often have limited user input capabilities, or none at all, and they may be installed in hard-to-reach locations.

We propose a new user-assisted protocol that provides a one-pass secure configuration procedure for wireless displays, connecting them simultaneously to a cloud service and to the wireless access network. The user simply scans a 2D bar code (QR code) on the freshly installed display through a camera phone, which takes the user to the management-service web page on the phone's web browser. The display is automatically added to the user's or the organization's managed display assets, and wireless access for the display is enabled.

The simplicity for the user is achieved thanks to a long-term trust relation between the user's organization and the cloud-based management service. For example, the user's employee or university would choose a specific cloud service for managing its displays. Home users could enable a similar service by turning it on in their wireless router. Our technical solution makes creative use of standard technology including RADIUS and EAP. We implement our own EAP method between the display and the management server; no software or hardware changes are needed on wireless access points or other network devices. Even in large organizations, minimal changes in the network configuration are typically sufficient to create the long-term trust relation with the management service. Displays that implement our protocol do not require any pre-existing relation with either the user or the management service.

RELATED WORK

Cloud-connected public displays have been suggested by several authors. For example, Lindén et al. [26] proposed a cloud-based web-application platform for deploying interactive services on public displays, Kubitz et al. [25] designed a system for personalized cloud-based applications on public displays, and there have been other cloud-connected display systems as well [27,31,37]. However, the previous work does not give much attention to how the displays are configured for Internet access and associated with the cloud services for secure communication.

As explained in the previous section and as noted by Roy and Trevor [39], this task of secure bootstrapping can be challenging, especially when the displays are wireless and need to establish a secure connection with appropriate wireless access points (APs). The main concern is the so-called evil twin attack, in which an attacker lures the potential victim to connect to a rogue access point by advertising the same network name (SSID) as a legitimate one. The attacker might then redirect the user to well-engineered phishing pages [15] and steal sensitive credentials or information.

At first, it may seem that the various device-pairing schemes from the literature provide ready solutions. However, these solutions are mainly intended for personal devices that are in the physical proximity of each other. In public spaces and in

the enterprise environment, the access points and especially the management server are often physically inaccessible to the user. This causes problems with pairing schemes based on physical proximity [11, 12, 28, 36] as well as visual or audio out-of-band channels. The Seeing-is-Believing [29] and Loud-and-clear [17] systems provided the important lesson, however, that the visual and audio channels can be used for verifying cryptographic data as the human user will easily detect spoofing attacks, unlike in electronic communication.

Kindberg et al. [23, 24] described a unidirectional network authentication technique for detecting evil-twin access points based on the Rivest and Shamir's Interlock protocol [35] and verification data shown on a public display or, alternatively, on a public web site. Their solution has many of the same components as ours but it has different security goals and assumptions; in particular, the display is just an auxiliary device and is already securely connected to the wireless access point.

As noted by Bauer et al. [10], out-of-band channels can be cumbersome and impractical for a secure connection between a wireless client and an AP. Instead, they proposed a one-way authentication method of APs that defines a new extension, called EAP-SWAT, to the 802.1X Extensible Authentication Protocol (EAP) [9]. In their solution, the client is supplied with a self-signed certificate from the AP on the first connection. The security of EAP-SWAT thus follows the *leap-of-faith* or *trust-on-first-use* model. The protocol is thus vulnerable to a man-in-the-middle attack on the first connection attempt when a rogue certificate may be presented to the client.

We were indeed inspired by the use of EAP by Bauer et al. There exist other EAP authentication methods such as EAP-TLS [38] that allow strong bidirectional authentication between a connecting client device and an access point. However, these EAP authentication methods were designed for devices that have pre-configured cryptographic authentication credentials like client certificates, or where the user can easily input a password or some other shared secret.

Establishing a secure connection between a display and an appropriate wireless AP is only a part of the problem. The other equally challenging task is to connect the display securely with the appropriate management and content servers. The closest widely deployed technology to the cloud-managed displays considered in this paper is Google Cloud Print [1], where smart printers are first connected to a wireless network and then to a cloud service. Thereafter, they can be accessed from any device with Internet connectivity. Like our displays, the printer is responsible for connecting to the cloud service, which solves the connectivity problems caused by firewalls and network address translation. Indeed, our bootstrapping solution, which was designed with cloud-managed displays in mind, could also be used for the initial configuration of such cloud-managed printers.

Naturally, manually provisioned device-specific credentials, such as usernames and password, can also be used for the cloud connection. Kerberos [30] is a typical example of such a protocol that has been used for authenticating devices in enterprise networks. However, the provisioning of creden-

Figure 1. System architecture.

tials is inconvenient if we expect the displays to be deployed ad-hoc by users rather than by system administrators. The 3GPP SIM-based GBA [7] solves the provisioning problem by using the cellular-network SIM card as the authentication credential. GBA can be a part of the solution, but it might not be pragmatic to assume that all wireless displays would be configured with a SIM-card for the bootstrapping alone.

To summarize, it is clear that the existing technologies can provide many ideas and inspiration, but new solutions for securely bootstrapping displays are still needed.

CLOUD-CONNECTED UBIQUITOUS DISPLAYS

This section provides a high-level description of our secure display bootstrapping solution. We first introduce the reference architecture and the design principles behind it, then discuss the intended user experience and security objectives.

Before proceeding further, we would like to emphasize the key aspects of our solution: the displays are administered by a *cloud-based management service*, show *web content* and access the Internet through a *wireless network*. Such displays may have *no usable user input method*; they are purchased or redeployed with *no pre-configuration* and are set up routinely by *untrained persons*. The displays need to be *associated securely* with both a *management service* and the *wireless access network*; they also should be as *self-configuring* as possible.

Reference architecture and design principles

The system architecture, induced by the key aspects introduced earlier, is illustrated by Figure 1 and comprises five major elements. On the one hand, a *ubiquitous display* – equipped with a wireless interface – is installed in a public (or private) space to provide certain services. An *enterprise wireless access point* is connected to the Internet and allows the display to access such services. It supports enterprise-level authentication and authorization, i.e., through the organization's own authentication server or an external trusted server. On the other hand, two server-side components are deployed in the cloud infrastructure. The *management server* is the core element of our proposed system and is involved in the whole lifecycle of the display. It takes care of the different aspects related to the initial configuration (including the connection to the Internet) as well as the authorization and management (i.e., monitoring and control) of the displays. Finally, a *content server* provides the actual service to the display for its intended purpose. For instance, a display installed at a bus stop could be instructed to show the public transport

timetable as well as location-based advertisements. Finally, the user's *mobile phone* bridges the local network with the cloud infrastructure in order to accomplish the initial secure configuration of the ubiquitous display. The mobile phone is assumed to have a camera and to be connected to the Internet, for instance, through a 3G or 4G technology.

The management server itself consists of several modules, each addressing a specific problem. The *bootstrapping module* authorizes users to perform the initial registration and configuration of the display. Specifically, it establishes a secure connection between the display, links it to a specific user account, and enables selecting the content that is initially shown. For instance, ubiquitous displays installed in a university might show the course timetables and the list of seminars, but may not show advertisements. The *monitoring and control module* allows users to supervise their registered display assets. They can inspect the content shown by the display and reconfigure it when needed. As ubiquitous displays are usually left unattended after the installation, the monitoring and control module is essential to troubleshoot issues related to both misconfiguration and hardware or software malfunctions [13].

The displays in our experimental implementation show web content and, more specifically, HTML5. This follows the model of many existing public display systems [26]. The application software for showing HTML5 content is a full-screen web browser; as a result, the software development costs are minimal and almost any display device will be compatible with minor changes. While our architecture does not in any way require web content, and other content-viewers could be launched, HTML5 seems the most practical and versatile choice as a standard content format for the foreseeable future. It enables us to freely combine cloud services, display hardware, and content from different vendors. We also exploit web-based technologies for the configuration and management of the displays. This is particularly useful for the mobile device, which does not require the installation of yet-another-app.

Both the display and its content need to be protected from unauthorized access. This is true for private content, but even more so if the display is at a public location. Thus, a secure association will need to be set up between the display and the management and content servers in the cloud. Furthermore, the displays need to have access to the Internet. The access network is typically a local WiFi network because of its low marginal cost for connecting any number of devices. WiFi

does, however, require access credentials such as a username and password. One of our major design goals is to establish the wireless access quickly on demand with minimal user interaction. Thus, we have to automate the network selection and credential configuration. We do not currently support communication over open wireless networks because it would be difficult for a self-configuring device with no user input to know whether the access is allowed. Successful authentication is an unambiguous indication that access is permitted. Of course, a suitable user interface could be added for selecting an open wireless network. We are also working on a version of the protocol for cellular data connection, but the focus of this paper is on displays that connect to the Internet over an access-controlled local network.

A fundamental assumption behind this work is that electronic displays will become commodity products that benefit from being self-configuring. When purchased or moved to a new location, they will not have any pre-configured information such as a management-server address, wireless-network name, or user accounts and keys. Moreover, the display deployment will be a routine task that should not require complex procedures or special training. Eventually, electronic-paper displays could be placed almost as casually as paper printouts and posters are fixed to walls today. Of course, since the content on the electronic displays can be updated online, much of the work associated with printed paper can be avoided, but anyone should nevertheless be able to install them.

User experience

The setup of a new display consists of the following phases.

1. After the display is powered up for the first time, it shows a QR code along with an organization name next to it.
2. The user scans the QR code with the built-in camera of a mobile phone and opens the received URL on a mobile web browser. The management server web page is opened.
3. The web page prompts the user to enter a username and password for an account on a management server. This step only takes place once in a while, after which the user stays logged in.
4. The user enters a description string for the display, takes a photo of its location and/or geotags it through the phone.
5. The user may select the content (i.e., a web application) for the display, or leave that to be done later.

From the user's point of view, the only requirement is the prior knowledge of the user account on the management server, to which the display will be registered. Most mobile phones have a camera, a web browser, and support the scanning of URLs from 2D bar codes. Thus, almost any modern phone can be used, and no new app needs to be installed.

The wireless network is discovered automatically. Sometimes, the display is in the range of multiple wireless networks that could be used to connect to different management servers. In that case, the display shows (or cycles through) multiple bar codes and the user selects one (in step 2) based on the familiar organization name.

Bootstrapping security

The goal of our bootstrapping protocol is to *enable the secure deployment of display devices* (i) *without any user input to the devices*; and (ii) *without per-device configuration changes to the local network*. The display may have just been unboxed and lacks any knowledge about the user, network or management server address. All the configuration information, including per-device keys, will be set up automatically. We achieve this automation by building on several existing standards and technologies in a novel way.

As already indicated, two distinct security associations are needed: one between the display and the access network, and one between the display and the management service. These security associations will be implemented by establishing shared secret keys, which are then used to protect the wireless and end-to-end communication, respectively, with standard security protocols. In a traditional system, these two security associations would be set up manually for each display, requiring the user to enter the usernames and keys or passwords (or long-term secret keys) into the display. Our bootstrapping protocol sets up these credentials automatically. It also ensures that each display has different credentials, so that the security of one display device does not depend on another.

In order to design a key-establishment protocol for the displays, we need to understand the security requirements precisely. In the following, the word *authorization* is used rather than authentication to emphasize that the display and management server do not reliably know each other's names or identifiers (they have no certificates) and that the trust relations are unidirectional, not symmetric. The two main security objectives are the following.

Authorizing a management service in the cloud to control the display. To establish a security association between the display and the management server in the cloud, we use an out-of-band channel provided by the user and mobile phone. Specifically, *the URL in the QR code contains a freshly-generated random number, the knowledge of which enables the management server to compute the keys that allow it to control the display*. The authorization is linked to a specific user account on the management server when the user logs in.

Authorizing the display to access the Internet. Next, we come to establishing the security association between the display and the wireless access network. For this, we need to introduce our second important architectural decision: *there is a pre-established trust relation between the management server and the wireless access network*. More precisely, the management server is authorized to grant devices Internet access through the wireless network. Thus, after the association between the management server and the display has been established with the help of the phone, the management server tells the wireless network to enable Internet access for the display.

The pre-established trust relation between the management server and the wireless network is a strong assumption. However, let us compare with the alternative. Without this relation, wireless network access would have to be manually en-

abled for each display device. Since we may need to revoke access for individual displays without rekeying all others, a separate account would have to be created for each display. Certainly it is far less administrative work to establish once the trust relation between the display management server and the wireless network than to create such individual accounts for each display. Moreover, as will be explained later, this can be done using standard protocols, without software or hardware upgrades at most wireless networks. As a result, the user will see the wireless network connection enabled almost “magically”. Therefore, we consider the unusual setup of the trust relations one of the major contributions of this paper.

Optionally, the user could tentatively link new displays to an organizational account even without logging in, so that the configuration process can be completed later by a logged-in user. Nevertheless, approval by an authenticated user is needed at some point to avoid misuse of the wireless network for free Internet access.

Sometimes, the content shown on the display may be private, which gives us a third security objective.

Authorizing the display to receive the content. The management service does not serve the content itself but redirects the display to the actual content pages at another address. It would be possible to derive keys for the content server from the key shared between the display and the management service. Such key derivation is used, for example, in the Generic Bootstrapping Architecture (GBA) of the 3GPP specifications [7]. Nevertheless, we found it more practical to decouple the content application completely from the management server. Some of our example applications achieve this third security objective by performing another round of QR code scanning with the phone after the management server has redirected the display to the content pages. That way, the content server takes care of its own security and does not even need to know how the display arrived to its web pages.

COMMUNICATION PROTOCOLS AND TECHNOLOGIES

The components of the proposed system are interconnected through a few key communication protocols and technologies, which will be described in this section.

The security of the bootstrapping process is accomplished through the Remote Authentication Dial In User Service (RADIUS) protocol [34]. RADIUS is commonly used for the centralized management of authorized network user accounts in secure enterprise wireless networks, as part of the WPA2-Enterprise protocol suite, and can also be used for controlling access to wired local-area networks. RADIUS has been specifically designed to support a large number of users and devices. The access points in enterprise networks are configured to connect to a RADIUS server, which performs the user authentication for them. The authentication is tunneled through the access point and other network infrastructure, so that only the client device and the RADIUS server need to be aware of the details. Moreover, RADIUS servers are able to route access requests to other, remote RADIUS servers, thus enabling roaming agreements for wireless access.

We make use of this capability so that the cloud-based display

management service in our system acts as a remote RADIUS server. Organizations that use the management service need to route the relevant access requests to this remote RADIUS server, thus delegating some network-access decision to it.

The tunneling of authentication between the wireless clients and the RADIUS servers follows the Extensible Authentication Protocol (EAP) [9] specification. EAP itself does not perform any authentication; rather, it allows the flexible definition of new client authentication methods, which may be tailored to specific application scenarios. In our proposed system, we indeed define a new EAP method for ubiquitous displays that is detailed next.

The Secure Display Initialization EAP method

We have designed a new EAP method called Secure Display Initialization (EAP-SDI) as part of our proposed system. We have made creative use of the standards, and EAP-SDI differs from most EAP methods in two significant ways. First, it not only authenticates previously registered clients but can also register new ones with the user’s help. Second, clients use this method to probe for networks that support our display management scheme.

From the security perspective, the EAP method implements a Diffie-Hellman key exchange [16] between the display and the management server. Diffie-Hellman is convenient for registering new devices because it can create a secure association (shared secret key) between two entities without authenticating them. We then rely on user’s action to ensure that the right two entities are associated. The user, with the phone, acts as an out-of-band channel that prevents man-in-the-middle attacks against the Diffie-Hellman key exchange. From then on, future authentications to the same wireless network will use the established key and will not require user interaction.

The initial EAP-SDI authentication exchange is shown in Figure 2. The ubiquitous display and the management server communicate with EAP encapsulated in RADIUS, as defined in the IEEE 802.1x standard [22]. The access point initiates the message exchange that ultimately results in a success or failure. Between these steps, it acts as a pass-through device that simply encapsulates EAP requests to and decapsulates responses from the corresponding RADIUS [34] messages. To save space, the diagram does not show the local RADIUS server, which routes the requests and responses to the management server (i.e., the remote RADIUS server) as illustrated by Figure 1.

For the 802.1x authentication, the display needs a Network Access Identifier (NAI) [8]. The ubiquitous displays that implement our protocol will have a NAI of the form UDID@ubidisplay.org, where a unique display identifier (UDID) is concatenated with the well-known domain name. The UDID is the Ethernet address of the wireless interface in hexadecimal format. The well-known suffix @ubidisplay.org enables the local RADIUS server to recognize display devices and to route their access requests to the chosen management server.

The user-assisted registration and configuration of the display has the following steps:

Figure 2. Message diagram for the Secure Display Initialization EAP method (EAP-SDI).

1. The display begins by sending an EAPoL-start message to the access point to initiate the authentication procedure. The access point responds with an EAP-Request for the identity of the display. The display submits its NAI to the access point.
2. The access point decapsulates the NAI from the EAP-Response message, encapsulates it in a RADIUS Access-Request message and forwards it to the local RADIUS server, which recognizes the well-known NAI domain part and routes it to the management server.
3. The management server composes a RADIUS Access-Challenge message that contains the server's cryptographic key-exchange data and additional information: a session ID (SessID) for this authentication exchange, the server's Diffie-Hellman public key g^x , a random server nonce N_{ms} , the management-server base URL, and the local organization name. This data, like all EAP-SDI messages, are in the JSON format to enable easy parsing and extension. The access point receives the RADIUS Access-Challenge and encapsulates its content
- in an EAP-Request message, which is then sent to the display.
4. Upon receiving the EAP-Request, the display responds with an EAP-Response that contains the display's cryptographic key-exchange data and additional information: the session ID (SessID), the client's Diffie-Hellman public-key g^y , a random client nonce N_{d1} , and additional information that will help the user later identify and manage the display (e.g., manufacturer, model and serial number). The access point forwards the EAP-Response to the management server.
5. The display also generates a second random client nonce N_{d2} , the knowledge of which will authorize the management server to control the display. The display then calculates three secret keys: the shared key material $K_{dh} = h(N_{ms}, N_{d1}, g^{xy})$ as a hash of the nonces and the Diffie-Hellman shared secret, the authentication key $K_{MAC} = h("MAC", k_{dh})$, and the master key $MK = h("MK", N_{d2}, K_{dh})$, which will be the long-term shared key for the display and management server.

The display then composes a URL from the base URL of the management server and the following query string parameters: the session ID (SessID) of the authentication exchange, the second random client nonce N_{d2} , the unique display identifier (UDID), and a message authentication code (MAC) of these values computed with the key K_{MAC} . This URL is converted to a QR code and shown on the display. The name of the organization is also shown to help the user identify the correct wireless network and management server.

6. At this stage, the user is responsible for completing the authentication exchange. The user scans the QR code shown by the display with a mobile phone camera. This delivers the URL, including the secret nonce K_{d2} , to the server. From the data received in the previous EAP message and the URL, the management server is able to compute the shared keys K_{dh} , K_{MAC} , and MK . It verifies the message authentication code in the URL and continues further only if it is valid.

The user then logs into the management server on the phone's web browser, unless he or she is already logged in. The login links the display to the user's personal or organizational account in the server. The user may enter a textual description of the display, take a picture of its location and/or geotag it through the mobile phone.

7. The management server then derives a Pairwise Master Key $PMK = h("PMK", N_{ms}, N_{d1}, MK)$ from the long-term master key MK . This PMK may be updated in future authentications while MK remains the same as long as the display is controlled by the management server. The server sends the PMK to the access point in a RADIUS Access-Accept message.
8. Upon receiving the RADIUS Access-Accept message, the access point sends an EAP-Success message to the display. The display also computes the PMK value. It then starts the 4-way handshake of IEEE 802.11 to complete the secure association with the access point. From there on, the protocol does not differ from any other wireless network authentication.

The display may need to reauthenticate to the network, for example, when the access point is reset and forgets the PMK, or when the display detects another access point with a better signal. In such future authentications, the master key MK will be reused, but the nonces N_{ms} and N_{d1} will be new. Thus, a new pairwise master key PMK will be computed. The authentication protocol will be a slightly abbreviated version of the initial authentication described above: instead of the Diffie-Hellman keys, only new nonces are exchanged, and no QR code is shown. The user will not even notice that the reauthentication takes place. Only when the user deletes the display registration from the management server or when the display device is reset to default settings, the display will again show a QR code and user assistance will be needed.

Access point selection and other details

In the following, we will cover some aspects of the display deployment and EAP-SDI that have not yet been discussed.

First, there could be more than one wireless network in the range of the display when it is powered up for the first time. Somehow, it needs to know which one to connect to. In our solution, the display scans all access points in its range that support the WPA2-Enterprise mode. It then attempts to connect to them in a round-robin fashion with the EAP-SDI method. Networks that do not support EAP-SDI respond with a NAK and are not contacted again after the first connection attempt. Each connection attempt to a network that does support EAP-SDI results in a different QR code shown on the display. The user is then responsible for selecting a QR code based on the organization name shown on the display. The management server and display will remember previous nonce values for a few minutes, so that the user has plenty of time to complete the authentication and display registration.

Second, if the display does not find any APs that support the new EAP method for display bootstrapping, or fails to successfully connect with any one of them, it will eventually enter an energy-saving mode and will need a button press, infrared remote-control signal, or some other user action to wake up and resume the authentication attempts.

Another question is how the EAP-SDI method fits into the small-office and home environments where authentication is more commonly based on a shared passphrase (WPA2-Personal) than on the 802.1x protocol. There are several alternatives to maintaining a local RADIUS server. First, some access points have the local RADIUS server built in, so that the routing can be set up in the access-point control panel. Second, access points may be configured to connect to a commercial cloud-hosted RADIUS server, which then plays the role of the "local" server in our protocol. Third, a secondary access point may be set up or an existing access point may be configured with a secondary network name (SSID), so that this second network is only used for displays and all access requests to it will be sent to the same remote RADIUS server.

Display operation

After the completion of EAP-SDI, the display will be registered at the management server. Since the display now has Internet connectivity and knows the management server base URL, it can connect to the server to retrieve further instructions. Our implementation uses HTTP redirection from a display-specific initial URL to a content service as the only control mechanism. However simple, such interfaces would need to be standardized. In our example applications, TLS without client authentication provides sufficient security. When the client needs to be authenticated, that could be done with a key derived from the master key MK .

After the initial configuration and troubleshooting, continuous monitoring is necessary to ensure that the displays are still functioning in the intended way, and that any unused hardware assets will be reclaimed. Also, additional control functionality may be needed to remotely recover displays that are stuck or need to be redirected to a new application without bootstrapping them again. We have implemented the monitoring and control through the Virtual Network Connection technology [33, 37]. Other remote display solutions can provide equivalent features. However, depending on the type of

Figure 3. The different pages shown by the display configuration wizard: (a) greeting and prompt for credentials; (b) display description; (c) service selection; (d) landing page after completion of the configuration. (e) Example of smart sign realized with an e-ink display.

the ubiquitous display, running a full-fledged remote desktop server might be an overkill due to resource constraints such as network bandwidth and battery power. Therefore, a lightweight custom management protocol may be needed for controlling displays. For web-based ubiquitous displays, this protocol could be embedded in a browser plugin; another option would be to use IFrames and Javascript. The reliability of such solutions remains an open question though. An advantage of VNC is that it is able to recover most problem situations by restarting the full-screen web browser or by rebooting the display device.

EVALUATION

In this section, we describe a prototype implementation of our proposed system, the user study that was conducted on the prototype, and also the results of a security analysis.

Prototype implementation

Our implementation involves software running at the different system components illustrated by Figure 1. At the ubiquitous display, we extended the `wpa_supplicant` package [6] to implement the new EAP method, EAP-SDI, in compliance with RFC 3748 [9]. Furthermore, we developed a simple python daemon that manages the access to the network and updates the content of the display after a successful authentication. This daemon is also responsible for the access point selection and re-connection strategies discussed earlier. Our implementation has been tested on Ubuntu Linux personal computers; however, it can easily be ported to Linux-based embedded systems such as the Raspberry Pi and Android-based devices.

In order to have conditions similar to those in practical deployments, we set up an enterprise wireless network with a local RADIUS server, as well as the remote RADIUS server hosted at the management server. The local RADIUS server is based on the FreeRADIUS open-source implementation [5]. The server was configured to proxy authentication requests belonging to the `ubidisplay.org` domain to the external RADIUS server hosted by the management server. The latter was realized through the `hostapd` package [2] that embeds a minimalistic self-contained RADIUS server. We added support for the EAP-SDI method to `hostapd`.

We employed python on the server side for the web application through which the mobile phone performs the initial configuration of the display. The web application exploits HTML5 and the JQuery Mobile framework [3] to provide a mobile-friendly, responsive and rich interface. This web application is organized as a “wizard” in order to guide the user through the different phases involved in the configuration of the display (Figure 3). The wizard is started as a result of the user scanning the QR code and opening the corresponding URL on a browser. The web application first greets the user and shows some information about the ubiquitous display, such as manufacturer and model. Username and password are prompted if the user does not already have an active session on the browser (Figure 3a). Otherwise, the user can directly proceed to enter additional information about the ubiquitous display. Such information may include the location and the description of the display. To this end, the web application allows the user to geotag the display through the GeoLocation API. Additionally, the user may take a picture as a visual description of the display through the `getUserMedia` API (Figure 3b). Afterwards, the wizard shows a list of services that the user is allowed to start on the display (Figure 3c). A landing page is shown upon completing the process and the user can logout if no other displays need to be configured (Figure 3d).

We did not implement content services. However, for the sake of the evaluation, we have realized two different services as representative instances of those used for real applications. The first one simply displays a video or a playlist from YouTube, and is meant to be used for advertisements or for bulletin boards. The second service shows information about a given room; it could be used, for instance, for applications involving smart signs, such as office door signs (Figure 3e).

We also implemented a rudimentary display monitoring and control module based on the VNC protocol. Since the displays are often behind a network address translator, we adopted an approach based on a reverse connection. Here, the management server keeps listening for connections initiated by the monitored displays. Such connections are triggered by the network configuration daemon after the successful authentication of the display.

User study

To evaluate the proposed system, we performed a user study involving people employed at our institution as well as external participants. In selecting people, we aimed at obtaining a wide spectrum of potential users. We were also particularly interested in people who had little experience with information and communication technologies. To this end, we looked for participants among not only students but also other staff including secretaries, janitors, and facility managers. In total, 23 people participated to the user study; 15 of them were female and their age ranged from 21 to more than 60 years.

We used different methodologies for our user study. We first exploited controlled observation. To this end, we set up a room with a Buffalo Air Station WZR-N300 access point and a 46" Sony BRAVIA TV connected to a personal computer. All the necessary network configuration was performed before starting the study. Participants were invited to sit on a couch in front of the display; they were then provided with a short introduction to the study and a brief description of the task to be performed. Specifically, participants were instructed to configure the display by scanning the QR code and then to complete the configuration process through the wizard described in the previous section. We did not provide many details about the process so that the participants had to rely on the instructions shown by the phone; the users were anyway invited to ask for clarifications as needed. In the evaluation, we assumed that the users were configuring a display for the first time; as a consequence, they had to login with a username and password which we provided. The users could also freely enter the information about the display in different ways. For instance, they could either provide a textual description or take a picture; similarly, they could enter some text for the location of the display or geotag it through the phone. We provided the participants with a Samsung Galaxy S3 phone in which we had installed a QR code scanning application (i.e., QR Droid [4]) and the Google Chrome browser. However, we allowed the participants to use their own phone, if the software requirements were satisfied. The users were encouraged to express their thoughts according to the think-aloud protocol; the conversations between them and the research team were recorded for further processing and analysis.

After the evaluation, the participants were asked to fill an online questionnaire. The questionnaire covered aspects related to both the users' feelings about the proposed system and their previous experience with related technology, such as QR codes and smart screens. In evaluating users' satisfaction, we used a 5-point Likert scale in which the value of 1 corresponds to "Strongly disagree" and 5 to "Strongly agree".

Findings

Here, we summarize the results of our user study.

QR codes. The scanning process created most of the troubles for unexperienced users – eleven participants had not scanned QR codes before our user study. In one case the process took so long that the registration process timed out and it had to be started over. One user, instead, praised the QR code scanning process (*"that was smooth"*), probably because of the "live"

feature of the used software (it does not require to take a picture but automatically detects and opens QR codes). The QR code scanning process was perceived as reliable: the median of the ratings was 5. Indeed, the scanning was not affected by any issues, except for two experiments that were performed during sunset due to backlight; in these cases, users had to get closer to the TV in order to successfully scan the QR code.

Usability. Most of the participants felt that the process was easy: the median of the ratings in the questionnaire was 5. Users also found the process fast (comments included *"that's it?"* and *"I don't feel I did anything"*); indeed, it took less than 5 minutes to be completed on the average. The actual duration was largely affected by the fact that many users had never scanned a QR code before the study. With minimal training, the bootstrapping process could easily be completed within a minute or so. All participants except for three took a picture as a description of the display; all except for two geotagged the display through the phone. In three cases the username and password were typed incorrectly at the first try and had to be re-entered; besides, no major problems were encountered. However, some of the unexperienced users exited the applications running on the phone by accidentally pressing the Android "back" button. This was probably exacerbated by the design of the Galaxy S3 phone that does not show any icon for the touch controls next to the physical home button.

Security. When asked if they felt "[the system] was secure", the participants were actually neutral, with a median of the ratings equal to 3. On one hand, all users found the input of the username and password natural; it was clear that the authentication was necessary to perform the configuration of the display. On the other hand, some of the participants raised some concerns on the security of the QR codes. One user reported *"it is believed that QR codes are insecure and can make one's smart phone vulnerable to attacks"*. Interestingly, the participants did not realize that the QR codes shown by the display are dynamic and contain special "tokens".

Understanding. Most of the users, despite the initial introduction, did not seem to clearly understand the purpose of the system. One user explicitly mentioned *"I didn't get the whole point of this"*; another one *"I like the idea but it seems you never see someone using it in real life"*. What emerged is that users do not have a clear idea of ubiquitous displays, even though they might have seen them in malls and in bus and metro stops. For most users, a display is an output device of a personal computer; as such, it does not need to be configured. Furthermore, most of the users do not even have experience with smart TVs, whereas a few had used large-size TVs attached to set-top-boxes, media centers or gaming consoles. The latter group explicitly pointed out the inconvenience in performing advanced configuration. In the questionnaire we specifically asked about experiences of configuring the network connection. One user reported *"The configuration of the wireless network access was pretty cumbersome. It could not work reliably and did not even tell what the problem was. Had to use wired connection instead"*. Another participant said *"The most problematic aspect is typing text [...] The user interface is often clunky and not intuitive"*.

Security analysis

We have already explained the security design in the “Bootstrapping security” section. It remains to consider the attacks that could circumvent the stated security goals.

The Diffie-Hellman key exchange is known to be vulnerable to man-in-the-middle attacks. RADIUS has its built-in authentication, which protects EAP from external attacks, but the attacker could very well perform a man-in-the-middle attack on the wireless access network. To detect this, we use the out-of-band channel provided by the user and the mobile phone for verifying the Diffie-Hellman shared key (the MAC included in the URL is a function of such a key). In order to succeed in the man-in-the-middle attack, the user would need spoof the QR code. It is difficult to see how this would be possible unless the user is tricked into scanning the wrong display. This is not a serious possibility in the applications that we have considered. However, when displays are used for viewing confidential content, many aspects of their management become harder, and the question of the attacker tricking the user to authorizing the wrong display could arise for real.

The most serious threat against our protocol is the use of TLS and a web browser for connecting to the management server. As in all web connections, the user will have to check the server and therefore needs to know its name. Phishing and typo squatting attacks are possible: the attacker may set up a fake access point that serves a subtly different base URL to the display, and the user will need to detect it. The usual countermeasures apply here: user education and bookmarking the URLs. The user could avoid the attacks by first logging into the management server through a bookmark or a trusted portal page and only then scanning the QR code.

Another issue is that the out-of-band channel (QR code) is vulnerable to visual spying. The code contains the secret random nonce N_{d2} , which gives the management server the ability to compute the secret keys and, thus, the authorization to control the display. If the attacker is able to spy this nonce, it can hijack control of the display. This will lead to denial of service for the legitimate display owner and possibly to the posting of unwanted content on the display, which can be embarrassing if the display configuration takes place in front of an audience. In any case, the solution is to reset the display to factory settings and to start again from the beginning. Fortunately, the attacker needs to be physically present at the display location to spy the QR code and the hijacking attacks cannot be done remotely over the Internet.

The momentary hijacking of control over the display can be serious if it enables the attacker to install malware to the display. This is not the case for purely HTML5-based displays, but VNC is an unsafe technology in this regard. Remote access to the display device’s desktop will enable all kinds of mischief and unintentional blunders. For this reason, we see VNC as only a temporary control solution.

Finally, one possible concern is that some users in our study did not feel they had done anything to configure the display. Thus, they might not have a clear idea that, by scanning a QR code, they are authorizing Internet access for the display de-

vice. This is probably not a major security issue here, but an interesting observation on the general level: complex security configuration processes can be important as rituals that help the users realize the seriousness of their actions.

DISCUSSION

The solution described in this paper can be seen as a discovery protocol: the end result is that new devices, which have just been brought to the range of a local wireless network, are registered to and managed by the appropriate server. However, our technical solution is quite different. First, we have taken the modern cloud-service perspective without bringing our own agent (hardware or software) to the access network. We also do not depend on broadcast messages (except for the WiFi beacon) or a service directory for the discovery, but rather establish the association between the device and the online service directly with the help of the user.

From the security perspective, the usual order and direction of the establishing wireless network security is reversed. Normally, devices first authenticate to the wireless network to gain Internet access and then connect to an online service. In our bootstrapping mechanisms, the display is first associated with the online management service, which enables the Internet access for it. This requires the setting up of a trust relation between the management service but, from then on, saves the work of establishing new wireless network connections.

While the focus of this paper is on electronic displays, the same architecture should be capable of handling other devices that need to be connected to online services. Our solution makes use of the display’s capability of showing a URL as a two-dimensional bar code. Other devices with the ability to output a machine-readable URL can obviously also use our protocol without modification. These include all devices with a graphic display, printers, as well as all devices that have a near-field-communication (NFC) interface with NDEF tag emulation capability.

CONCLUSION

In this paper, we opened the investigation into the protocol aspects of ubiquitous wireless displays that are managed by cloud services. We expect such displays to eventually replace paper and ink in most applications and, thus, require a very simple and non-technical configuration process. Nevertheless, such a process needs to meet specific security goals. We have designed and implemented a new user-assisted protocol for the secure bootstrapping and management of wireless displays, and evaluated its usability and security. This solution provides a one-pass secure configuration procedure for wireless displays, connecting them simultaneously to a cloud service and to the wireless access network.

ACKNOWLEDGMENTS

This work was supported partially by TEKES as part of the Internet of Things program of DIGILE (the Finnish Strategic Center for Science, Technology and Innovation in the field of ICT and digital business) and by the US National Science Foundation (NSF) under grant CNS-1150192. The authors would like to thank Päivi Kinnunen for her help in the planning of the user study.

REFERENCES

1. Google cloud-based print service.
<http://www.google.com/cloudprint/learn/>.
2. hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator.
<http://hostap.epitest.fi/hostapd/>.
3. jQuery Mobile framework. <http://jquerymobile.com/>.
4. QR Droid.
<https://play.google.com/store/apps/details?id=la.droid.qr>.
5. The FreeRADIUS Project. <http://freeradius.org/>.
6. WPA Supplicant for Linux, BSD, Mac OS X, and Windows. <http://hostap.epitest.fi/wpa-supplicant/>.
7. 3GPP Organizational Partners. Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA). 3GPP Technical Specification 33.220, European Telecommunications Standards Institute (ETSI), December 2013. version 12.2.0.
8. Aboba, B., Beadles, M. A., Arkko, J., and Eronen, P. The Network Access Identifier. RFC 4282, Internet Engineering Task Force (IETF), December 2005.
9. Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and Levkowetz, H. Extensible Authentication Protocol (EAP). RFC 3748, Internet Engineering Task Force (IETF), June 2004.
10. Bauer, K., Gonzales, H., and McCoy, D. Mitigating evil twin attacks in 802.11. In *IEEE International Performance, Computing and Communications Conference (IPCCC)*, IEEE (2008), 513–516.
11. Bichler, D., Stromberg, G., Huemer, M., and Löw, M. Key generation based on acceleration data of shaking processes. In *UbiComp: Ubiquitous Computing*. Springer, 2007, 304–317.
12. Buhan, I., Doumen, J., Hartel, P., and Veldhuis, R. Feeling is believing: a secure template exchange protocol. In *Advances in Biometrics*. Springer, 2007, 897–906.
13. Clinch, S., Davies, N., Friday, A., and Clinch, G. Yarely: A software player for open pervasive display networks. In *Proceedings of the 2Nd ACM International Symposium on Pervasive Displays*, PerDis '13 (2013), 25–30.
14. Dementyev, A., Gummesson, J., Thrasher, D., Parks, A., Ganesan, D., Smith, J. R., and Sample, A. P. Wirelessly powered bistable display tags. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13 (2013), 383–386.
15. Dhamija, R., Tygar, J. D., and Hearst, M. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, ACM (2006), 581–590.
16. Diffie, W., and Hellman, M. E. New directions in cryptography. *IEEE Transactions on Information Theory* 22, 6 (1976), 644–654.
17. Goodrich, M. T., Sirivianos, M., Solis, J., Tsudik, G., and Uzun, E. Loud and clear: Human-verifiable authentication based on audio. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS)*, IEEE (2006).
18. Heikenfeld, J., Drzaic, P., Yeo, J.-S., and Koch, T. Review paper: A critical review of the present and future prospects for electronic paper. *Journal of the Society for Information Display* 19, 2 (2011).
19. Henzen, A., Ailenei, N., Van Reeth, F., Vansichem, G., Zehner, R. W., and Amundson, K. 32.4: An electronic ink low latency drawing tablet. *SID Symposium Digest of Technical Papers* 35, 1 (2004), 1070–1073.
20. Henzen, A., van de Kamer, J., Nakamura, T., Tsuji, T., Yasui, M., Pitt, M., Duthaler, G., Amundson, K., Gates, H., and Zehner, R. 13.2: Development of active matrix electronic ink displays for handheld devices. *SID Symposium Digest of Technical Papers* 34, 1 (2003), 176–179.
21. IEEE Std. 802.11 WG, Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications”, 1999. (Reaffirmed June 2003).
22. IEEE Standard for Local and Metropolitan Area Networks – Port-based Network Access Control, February 2010.
23. Kindberg, T., Bevan, C., O'Neill, E., Mitchell, J., Grimmett, J., and Woodgate, D. Authenticating ubiquitous services: A study of wireless hotspot access. In *Proceedings of the 11th International Conference on Ubiquitous Computing*, UbiComp '09 (2009), 115–124.
24. Kindberg, T., Mitchell, J., Grimmett, J., Bevan, C., and O'Neill, E. Authenticating public wireless networks with physical evidence. In *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WIMOB)*, IEEE (2009), 394–399.
25. Kubitza, T., Clinch, S., Davies, N., and Langheinrich, M. Using mobile devices to personalize pervasive displays. *ACM SIGMOBILE Mobile Computing and Communications Review* 16, 4 (2013), 26–27.
26. Lindén, T., Heikkinen, T., Kostakos, V., Ferreira, D., and Ojala, T. Towards multi-application public interactive displays. In *Proc. of the 2012 International Symposium on Pervasive Displays* (2012), 9:1–9:5.
27. Lu, Y., Li, S., and Shen, H. Virtualized screen: A third element for cloud–mobile convergence. *Multimedia*, *IEEE* 18, 2 (February 2011), 4–11.
28. Mayrhofer, R., and Gellersen, H. Shake well before use: Intuitive and secure pairing of mobile devices. *IEEE Transactions on Mobile Computing* 8, 6 (2009), 792–806.

29. McCune, J. M., Perrig, A., and Reiter, M. K. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *IEEE symposium on Security and privacy*, IEEE (2005), 110–124.
30. Miller, S. P., Neuman, B. C., Schiller, J. I., and Saltzer, J. H. Kerberos authentication and authorization system. In *In Project Athena Technical Plan*, MIT (1987).
31. Oat, E., Di Francesco, M., and Aura, T. MoCHA: Augmenting pervasive displays through mobile devices and web-based technologies. In *Proc. of the first IEEE Workshop on Developing Applications for Pervasive Display Networks (PD-Apps '14)* (March 2014), 506–511.
32. O'Hara, K., Perry, M., Churchill, E., and Russell, D., Eds. *Public and situated displays: Social and interactional aspects of shared display technologies*, vol. 2 of *Computer Supported Cooperative Work*. Springer, 2003.
33. Richardson, T., Stafford-Fraser, Q., Wood, K., and Hopper, A. Virtual network computing. *Internet Computing, IEEE* 2, 1 (January/February 1998), 33–38.
34. Rigney, C., Rubens, A. C., Simpson, W. A., and Willens, S. Remote Authentication Dial In User Service (RADIUS). RFC 2865, Internet Engineering Task Force (IETF), June 2000.
35. Rivest, R. L., and Shamir, A. How to expose an eavesdropper. *Communications of the ACM* 27, 4 (1984), 393–394.
36. Sethi, M., Antikainen, M., and Aura, T. Commitment-based device pairing with synchronized drawing. In *Proc. of PerCom '14* (March 2014), 181–189.
37. Simoons, P., De Turck, F., Dhoedt, B., and Demeester, P. Remote display solutions for mobile cloud computing. *Computer* 44, 8 (August 2011), 46–53.
38. Simon, D., Aboba, B., and Hurst, R. The EAP-TLS authentication protocol. RFC 5216, Internet Engineering Task Force (IETF), March 2008.
39. Want, R., and Perring, T. System challenges for ubiquitous & pervasive computing. In *Proceedings of the 27th International Conference on Software Engineering*, ICSE '05 (2005), 9–14.