Ubiquitous One-Time Password Service Using Generic Authentication Architecture

Chunhua Chen[1⋆], Chris J. Mitchell[2], and Shaohua Tang[3]

[1,3] School of Computer Science and Engineering
South China University of Technology
Guangzhou 510640, China
[1] chunhua.chen@mail.scut.edu.cn, [3] csshtang@scut.edu.cn
[2] Information Security Group
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
c.mitchell@rhul.ac.uk

Abstract. In this paper we exploit GAA to build a scheme that converts a simple static password authentication mechanism into a one-time password (OTP) system. The scheme employs a GAA-enabled user device with a display and an input capability (e.g. a 3G mobile phone) and a GAA-aware server. Most importantly, the device does not need to be user or server specific, and can be used in the protocol with no registration or configuration (except for the installation of the necessary application software). The system also fits well to the multi-institution scenario and hence enables the provision of ubiquitous and on-demand OTP services.

Keywords: One-time password, Generic Authentication Architecture, mobile security

1 Introduction

A one-time password (OTP) is a means of proving the identity of a user in which a password is only valid for a single authentication session or for a short time period. On-demand OTP systems (e.g. RFC 4226 [7]) typically involve a user and server-specific security token that shares a secret key with the server and that requires an initialisation process. If the token is compromised, lost or stolen, then an adversary may be able to use it to access the server, at least until it is revoked. Such schemes do not fit well to the multi-institution scenario, since a user will be required to possess a token for each server with which the user interacts. Of course, a multi-institution scheme in which a token is equipped

⋆ The author is a PhD student at the South China University of Technology. This work was performed during a visit to the Information Security Group at Royal Holloway, University of London, sponsored by the Chinese Scholarship Council and the Natural Science Foundation of Guangdong Province, China (No. 9351064101000003).

with a separate secret key for each of a number of servers is technically feasible; however, such a scheme may be difficult to deploy and manage in practice. These disadvantages increase the management burden and cost, and hence limit the use of OTP systems to security-critical applications.

Molva and Tsudik [6] originally proposed the use of a non-user-specific security token (card) for user authentication. The card shares a strong cryptographic key with an authentication server, and is used solely to provide a secure channel between a human user and the server. Since the token is not associated with any particular user, the scheme is resistant to token compromise. However, it requires every server to generate and securely distribute a key-bearing token to every user, which is likely to be a significant burden in practice.

To avoid these disadvantages, we exploit the Generic Authentication Architecture (GAA) [2] to build a scheme that converts a simple static password authentication mechanism into an OTP system. A mobile network which provides the GAA service plays the role of a Trusted Third Party (TTP). The scheme employs a GAA-enabled user device with a display and an input capability (e.g. a 3G mobile phone) and a GAA-aware server. During the user authentication process, an application-specific session key is bootstrapped between the device and the server using GAA. The device computes a one-time authenticator as a function of this bootstrapped key and a long-term secret password shared by the user and the server. This authenticator is submitted to the server to authenticate the user, instead of the long-term password. The authenticator will be valid only if it is computed using a valid session key. Thus, limiting the validity of a session key to a short time period or a single authentication session makes this a one-time scheme.

The rest of this paper is organised as follows. In section 2 we give a brief introduction of GAA. We describe the GAA-based OTP system in section 3, and discuss its advantages and limitations in section 4. In section 5 we provide an informal security analysis, and in section 6 we draw conclusions.

2 Generic Authentication Architecture

The information in this section is mainly derived from Holtmanns et al. [4].

GAA has been standardised by both the 3rd Generation Partnership Project (3GPP), and its North American counterpart, the 3rd Generation Partnership Project 2 (3GPP2). GAA exploits the UMTS authentication infrastructure (strictly, the UMTS Authentication and Key Agreement (UMTS AKA) protocol [1]) to enable the provision of security services, including key establishment, to third party mobile and Internet applications.

The UMTS AKA protocol provides authentication and key establishment using a long-term secret subscriber key (K), shared by a user device (e.g. a 3G mobile phone) and a mobile network. As a result of a successful UMTS AKA procedure, a pair of secret session keys is shared by the device and the network. These keys are CK, used for confidentiality protection, and IK, used for integrity

protection. We note also that, in the UMTS AKA procedure, a random challenge ($RAND$) is sent by the mobile network to the user device.

GAA consists of two procedures, GAA bootstrapping and Use of bootstrapped keys. GAA bootstrapping, also known as the Generic Bootstrapping Architecture (GBA), is the process by which UMTS AKA is used to set up a GAA master session key (MK) between a GAA-enabled device and a network, where MK is the concatenation of IK and CK. The network also sends a transaction identifier $B\text{-}TID$[1] that can be used to identify MK and its lifetime to the GAA-enabled device. Both the GAA-enabled device and the network cache MK, the lifetime of MK and $RAND$[2] for later use. The master session key MK is not bound to a particular application, and can only be used to derive application-specific session keys.

Use of bootstrapped keys is the procedure by which a GAA-enabled device employs the bootstrapped keys to secure its exchanges in an application protocol with a particular GAA-aware application server. Once the GAA-enabled device decides to engage in an application protocol with a particular GAA-aware server, it derives an application-specific session key (SK) from MK, as follows:

$$SK = \text{KDF}(MK, \text{GBAvariant}, RAND, \text{IMPI}, NAF\text{-}Id)^3$$

where KDF is a key derivation function. The device starts the application protocol by sending a request containing $B\text{-}TID$. The server fetches the same SK, the lifetime of SK, and other relevant information from the corresponding mobile network by forwarding the received $B\text{-}TID$ and its own identifier $NAF\text{-}Id$.[4] At this point, the device and the server share the same value of SK. It is important to note that SK is bound to a specific application protocol and a particular application server.

3 The GAA-based OTP Scheme

As discussed above, GAA builds on UMTS AKA to bootstrap application-specific session keys between GAA-enabled devices and GAA-aware servers, and which can be used to establish a secure authenticated channel. In this paper

[1] $B\text{-}TID$ is generated from the $RAND$ value and the mobile network name.

[2] Here and throughout $RAND$ refers to the random challenge sent as part of UMTS AKA.

[3] GBAvariant indicates where to store the bootstrapped keys. In the normal GBA (GBA_ME) case, the bootstrapped keys are stored in the mobile device, and not in the UMTS Subscriber Identity Module (USIM). The IP Multimedia Private Identifier (IMPI) is derived from the International Mobile Subscriber Identity (IMSI) [3]. $NAF\text{-}Id$ consists of the Fully Qualified Domain Name (FQDN) of the intended server and the identifier of the application protocol.

[4] Normally the network has to authenticate that the requesting server is the genuine owner of FQDN, which forms a part of $NAF\text{-}Id$. In GAA, it is assumed that a secure authenticated channel between the server and the network has been set up by some means.

we combine this secure authenticated channel with a long-term secret password shared by the user and the server to obtain an OTP scheme.

In the system, the following entities play a role:

- A user U.
- A client (e.g. a browser) C, used by U to access a GAA-aware server.
- A GAA-enabled user device T (e.g. a 3G mobile phone) with a display and an input capability. T must possess a means of communication with the client, e.g. as provided by a USB cable or a Bluetooth link. T must also be equipped with an application supporting the OTP service, which must be capable of accessing a GAA service.
- A GAA-aware server S.
- A mobile network provider (TTP) that provides the GAA service.

These entities are equipped with a variety of parameters and cryptographic keys. U is equipped with an identifier *username* and a *password* (pw), a weak secret shared with S. A long-term secret subscriber key (K) is shared by T (strictly its USIM) and its home mobile network provider as part of the subscription. We assume that S has the means to establish an secure authenticated channel (e.g. as provided by an SSL/TLS tunnel) with TTP as necessary to use GAA.

1. $T \leftrightarrow TTP$: *B-TID*, *MK*, *RAND*
   : and the lifetime of *MK* ($[T_{start}, T_{end}]$).
2. T : derives a session key *SK*.
3. T : computes $otp = f(SK, pw)$ and then
   : deletes *MK*, *SK*, and others values.
4. $U(C) \rightarrow S$: *B-TID*, *username*, and *otp*.
5. $S \leftrightarrow TTP$: *SK* and the lifetime of *SK* ($[T_{start}, T_{end}]$).
6. S : $T_{current} \in [T_{start}, T_{end}]$?
   : if so, S recomputes *otp* for authentication;
   : if not, S discards the request.

Figure 1: the OTP protocol

Figure 1 summarises the OTP protocol. We next give a more detailed description, referring to the step numbers given in the figure.

When U wishes to access S, U directs its client C to S. T triggers a GAA bootstrapping procedure with its home network (step 1). After successful execution of this process, the values *B-TID*, *RAND*, *MK*, and the lifetime of *MK* ($[T_{start}, T_{end}]$) are shared and cached by T and its home network. The client C also provides T with the FQDN of the intended server S and the identifier of the application protocol, and T uses them to construct *NAF-Id*. T derives a application-specific session key *SK* as (step 2):

$$SK = \mathrm{KDF}(MK, \mathrm{GBAvariant}, RAND, \mathrm{IMPI}, NAF\text{-}Id).$$

Note that *SK* is not specific to U, and cannot be used to authenticate U to S. However, *SK* is specific to S.

After derivation of SK, T uses U's password pw (which must be input by U at some point) to compute an authenticator otp as a function f of SK and pw (step 3), i.e.

$$otp = f(SK, pw).$$

The function f can be implemented in many ways. One possibility is to instantiate f using HMAC [5] based on a suitable cryptographic hash function. That is, otp could be computed as:

$$otp = \text{HMAC}_{SK}(pw).$$

After computation of otp, T deletes MK, SK, pw and other relevant parameters.

The server authenticates the user U by asking him or her to submit the values *username*, *B-TID*, and otp over the channel between the client and the server (step 4). Note that the user is not required to enter these values into the client, since they can be transferred electronically from T to C. To verify the received otp, S forwards the received *B-TID* and its own identifier *NAF-Id* to T's home network and receives back the same SK as available to T, the lifetime of SK, and other relevant information using the GAA bootstrapping key usage procedure (step 5). As discussed previously, otp is valid only if it is computed using a valid session key SK. The scheme requires that SK has a short validity period, and its lifetime must be set to be the same as for MK, that is $[T_{start}, T_{end}]$. Before recomputing otp, S must check whether or not SK is valid. This is achieved by checking whether or not the current system time of S (i.e. $T_{current}$) is within the period $[T_{start}, T_{end}]$. If not, SK is invalid and U will be rejected; otherwise, S can now use the received SK to recompute otp for verification. If the computed otp and the otp submitted by U match, U will be granted access (step 6). Note that we assume that the system time of the network and the server are synchronised with each other. Thus, if the lifetime of SK is chosen to be sufficiently short then the scheme gains the one-time property.

We observe that we can also arrange for otp to be valid for only one authentication session. To achieve this, S must check whether or not the received *B-TID* has been used previously. If not, the corresponding SK is valid. Note that this requires S to cache all the *B-TID* values it encounters.

4 Advantages and Limitations

Unlike previously proposed OTP schemes, the system does not require an initialisation process. That is, the GAA-enabled mobile device is neither user nor server specific, and can be used in the protocol with no registration or configuration (except for the installation of the necessary application software). This approach thus fits well to the multi-institution scenario. The system enables server-specific session keys to be generated using a single GAA-enabled device, where each such key can be used to help authenticate a user to the appropriate GAA-aware server. The GAA-enabled device thus acts as a non-institution-specific authentication token.

To implement the scheme, the user only needs a GAA-enabled device with a display and an input capability. This could be a 3G mobile phone with a valid subscription, and there are a very large number of subscription holders across the world. The approach thus has good scalability. Moreover, since a GAA bootstrapped session key is used in the computation of the authenticator, there is no need to generate and securely distribute a key-bearing token to every user. The scheme can be implemented and deployed as a standard GAA-based OTP system to enable the provision of ubiquitous and on-demand one-time password services.

However, in deciding whether to use this one-time password service, the server S must trust the mobile network provider not to compromise its long-term password by intercepting client-server communications and performing a brute force search (see also section 5). Such a trust relationship could be supported by a contractual agreement.

Finally, use of the scheme will incur the cost of using the GAA service. However, challenge-based and counter-based OTP schemes can be devised which reduce the use of GAA. In such a scheme, the GAA bootstrapped session keys are used for multiple login sessions, and hence, GAA bootstrapping is not needed for every login attempt. These schemes are more efficient and cost-effective, but require the management of GAA bootstrapped session keys.

5 Informal Security Analysis

The GAA-enabled user device must be trusted by the user, since it has access to the user's long-term password. We also assume that the information within the user device is protected against attack by an untrustworthy or compromised client platform. As discussed in Holtmanns et al. [4], it is an implicit assumption in any application of GAA that learning the subscriber key and/or MK by attacking UMTS AKA is not possible. Another assumption is that a secure authenticated channel between the server and the network has been set up by some means, which implies that the network has the means to authenticate the requester against the FQDN. As a result, an adversary can neither obtain SK by monitoring the link between S and TTP nor request the SK intended for S from TTP, since the adversary cannot claim ownership of the FQDN for this SK.

However, even if an the machine hosting the client C has been compromised, e.g. via installation of a keylogger, an adversary cannot re-use a captured otp, or use it to obtain the user's long term password. Re-use is prevented by the fact that, as stated in section 3, otp is only valid for a short time period. Also, otp is computed using a keyed one way hash function with a strong cryptographic key (the GAA bootstrapped session key); it is thus infeasible to retrieve pw from otp without knowing this session key. That is, A cannot succeed in an off-line dictionary attack against pw without knowing the corresponding SK or MK (using MK, A can derive SK).

The best strategy for A is to mount an on-line dictionary attack against pw. In such an attack, A guesses pw and uses a GAA-enabled phone to engage in the protocol (masquerading as the user) to test whether this single guess is correct or not. However, to defeat such an attack, S can choose to lock out a user after a fixed number of failed authentication attempts.

It is very important to note that TTP possesses the GAA bootstrapped session keys and must be trusted since, if it intercepts the authenticator, it could perform a dictionary attack to find the user's long-term password.

Note that the GAA bootstrapping keys and the user password should be deleted from the mobile device after the computation of otp. Moreover, one-time passwords are consumed on demand, and thus the only secret that needs to be securely managed is the user's long-term password, which is known only by the user and the server.

6 Conclusions

We have proposed a scheme that converts a password authentication mechanism to a one-time password system using GAA. Most importantly, the GAA-enabled device is neither user nor server specific, and can be used in the protocol with no registration or configuration. An informal analysis shows that the scheme is effective, secure, scalable and fits well to the multi-institution scenario. Hence, the scheme can be implemented and deployed as a standard GAA-based OTP system to enable the provision of ubiquitous and on-demand one-time password services.

We conclude by observing that the role of the GAA infrastructure in the novel scheme could be replaced by any other pre-existing security infrastructure (e.g. a Trusted Platform, in which Trusted Platform Module uses its certified public key pair), as long as the security infrastructure service provider is trusted not to compromise individual user passwords.

References

1. 3rd Generation Partnership Project (3GPP). *3G Security: Access Secure for IP-based Services, Version 9.3.0*, 2009.
2. 3rd Generation Partnership Project (3GPP). *3rd Generation Partnership Project, Technical Specification Group Services and Systems Aspects, Generic Authentication Architecture (GAA), Generic Bootstrapping Architecture, Version 9.2.0*, 2009.
3. 3rd Generation Partnership Project (3GPP). *Numbering, Addressing and Identification, Version 9.2.0*, 2009.
4. S. Holtmanns, V. Niemi, P. Ginzboorg, P. Laitinen, and N. Asokan. *Cellular Authentication for Mobile and Internet Services.* John Wiley and Sons, hardcover edition, December 2008.
5. H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104 (Informational), February 1997.
6. R. Molva and G. Tsudik. Authentication method with impersonal token cards. In *Proceedings of the 1993 IEEE Symposium on Security and Privacy*, pages 56–65, Oakland, California, USA, 1993. IEEE Computer Society.

7. D. M'Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen. HOTP: An HMAC-Based One-Time Password Algorithm. RFC 4226 (Informational), December 2005.