Royal Holloway
University of London

Information Security Group

Building general purpose security services on EMV payment cards

Chris Mitchell

Information Security Group
Royal Holloway, University of London
http://www.isg.rhul.ac.uk/~cjm

Royal Holloway
University of London

Information Security Group

Acknowledgements

- This is joint work with Chunhua Chen and Shaohua Tang (South China University of Technology).
- Work partly conducted while Chunhua Chen was visiting RHUL.

Contents

- Security infrastructures
- GAA
- UMTS-GAA
- EMV-GAA
- Applying GAA variants
- Conclusions

Need for infrastructure

- Just about any system using cryptography for security needs a key management system.
- This typically involves either:
  - setting up shared keys, e.g. between a server and multiple clients;
  - setting up a PKI by requiring clients to generate key pairs and obtain public key certificates from a CA.

Cost implications

- Setting up a new security infrastructure is a potentially very costly business.
- Distributing SIMs to all the users of a mobile phone network makes sense because of the sales volume – however, for other services the cost of such a solution becomes prohibitive.
- The alternative, widely used today, involves a combination of user passwords and one-way authenticated SSL/TLS – this approach has many, widely documented, vulnerabilities.

Infrastructure re-use

- Therefore appealing to find ways to build on existing security infrastructures.
- Two main motives:
  - increased security and relatively low cost for service provider;
  - extra revenue stream for infrastructure owner.
- This is already happening, e.g. through NFC-based credit/debit card emulations built into mobile phones.

Contents

Background

- The term *Generic Authentication Architecture* (GAA) has been developed within the mobile phone community.
- It refers to a standardised way of exploiting the mobile phone security infrastructure to provide general purpose authentication and key management services.
- The mobile operator acts as a TTP.
- We start by describing this architecture in general terms.

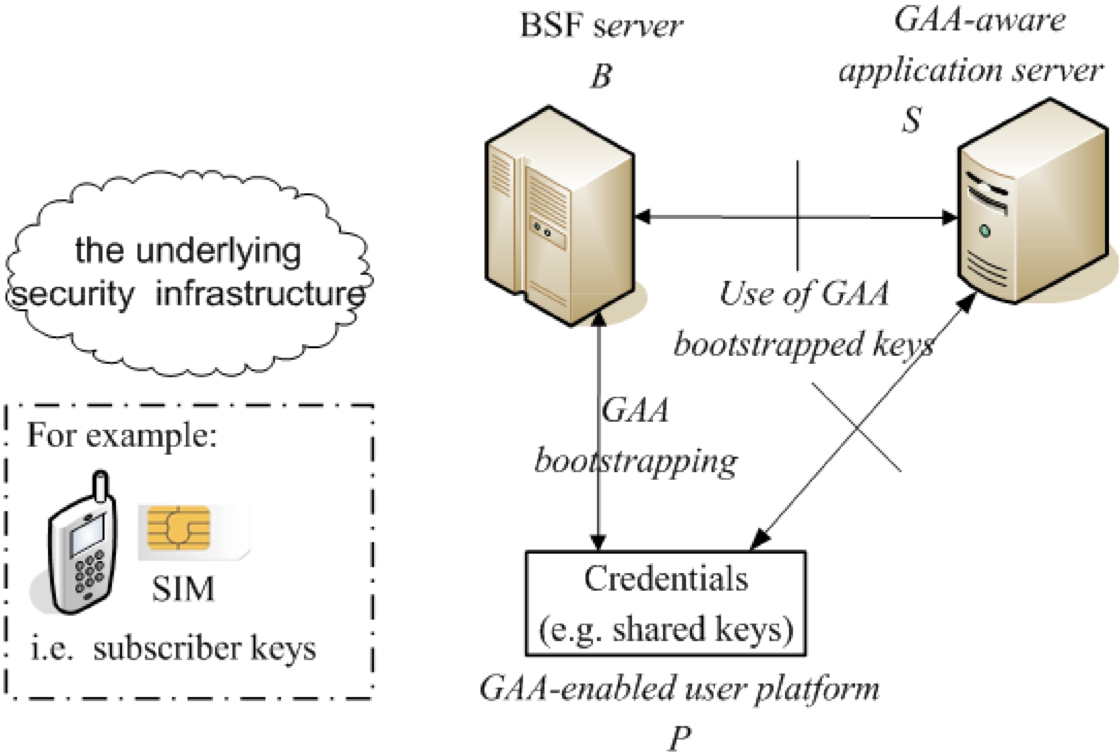
GAA roles

- The GAA architecture involves three roles:
  - **Bootstrapping Server Function (BSF)** – this is the TTP that provides the service;
  - **GAA-aware application server** – has trust relationship with BSF;
  - **GAA-enabled user platform** – has an existing security relationship (e.g. shared secret key) with the BSF.

GAA service

- GAA establishes an authenticated **application- and server**-specific secret key between the GAA-enabled user platform and an arbitrary GAA-aware application server.
- The user must have a mobile phone subscription.
- The target server must have a relationship with the GAA service provider.

GAA overview



GAA procedures

- Two main procedures:
  - **GAA bootstrapping** – Establishes a secret master key *MK* (and an identifier *B-TID* for the key and a key lifetime) between GAA-enabled user platform and the BSF.
  - **Use of bootstrapped keys** – Establishes an application- and server-specific session key *SK* between platform and server using *MK* [*MK* is not divulged to the server]:

$$SK = f(MK, \text{server-ID}, \text{app-ID}, \ldots)$$

where *f* is a key derivation function.

Key provisioning

- The GAA-enabled user device can calculate *SK* for itself.
- The GAA-enabled server is provided with *SK* by the BSF.
- Thus a secure channel between the BSF and the server is required.

Our goal

- GAA was designed specifically for use with the 3G mobile telecoms. security infrastructure (we call this UMTS-GAA).
- We show how to provide GAA-like services with other pre-existing infrastructures.
- As a result, any services built on UMTS-GAA can immediately be migrated to other security infrastructures.

Contents

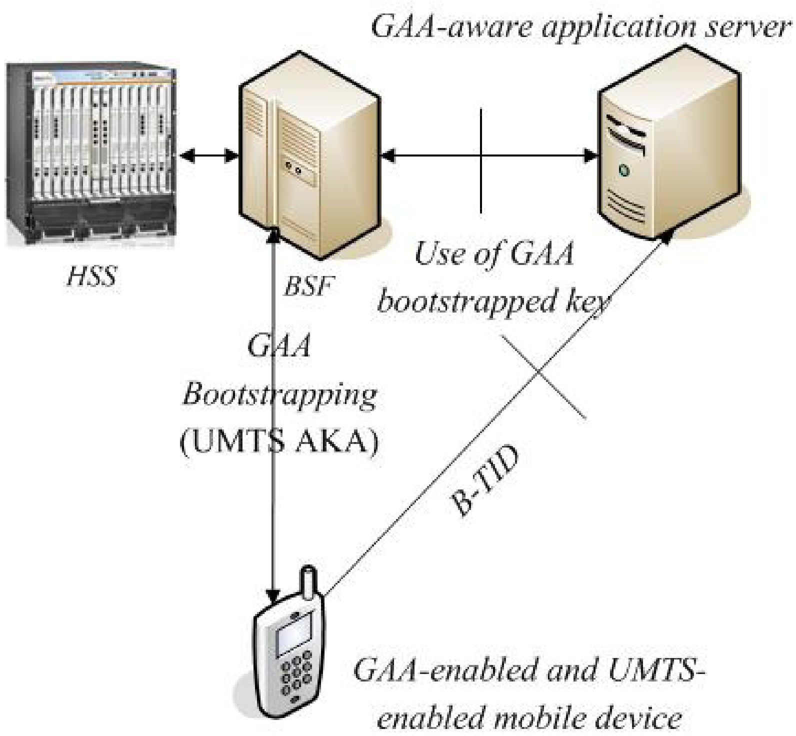
UMTS – background

- The UMTS security infrastructure (supporting mobile phone security) has the following roles:
  - **USIM** – smart card held by user (in phone);
  - **Home Subscriber Server (HSS)** – shares secret key with USIM, and operated by mobile phone service provider with whom user has contractual relationship.

UMTS-GAA

- In UMTS-GAA:
  - GAA-enabled user platform is a UMTS-enabled mobile device, with a USIM;
  - BSF connects to the appropriate HSS for the USIM (may be owned by same operator);
  - UMTS Authentication and Key Agreement protocol (UMTS AKA) is used to establish *MK* between GAA-enabled user platform and BSF (*MK* is concatenation of *IK* and *CK*).

UMTS-GAA



Session key derivation

- In use of bootstrapped keys:

  SK=*f*(*MK*, RAND, mobile-ID, server-ID, app-ID,…)

- RAND is the value used in the UMTS AKA protocol (functions as a random challenge in the protocol).

Contents

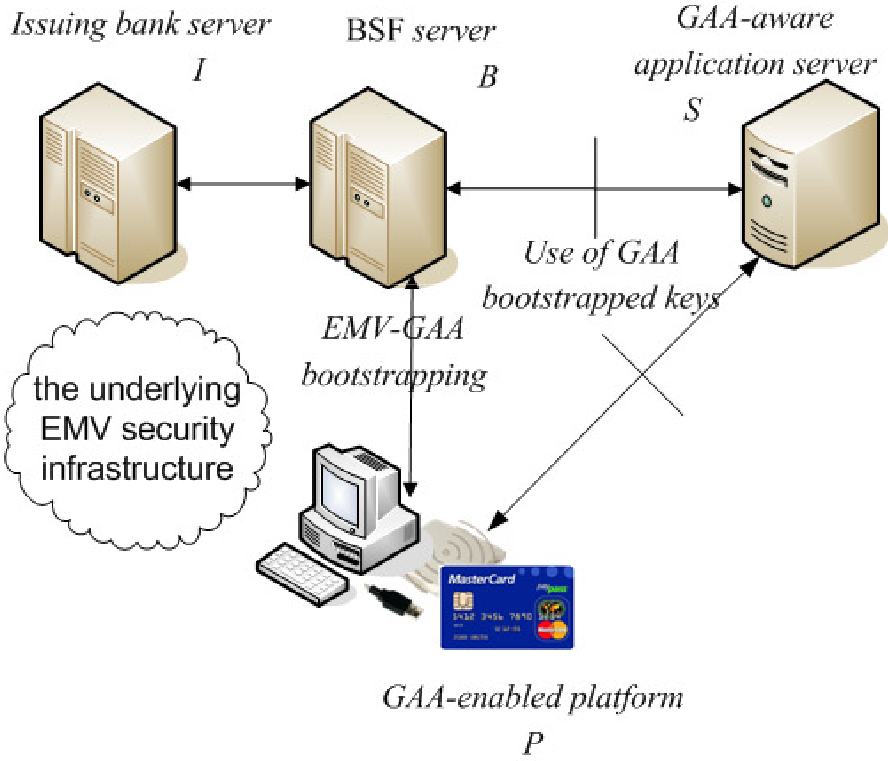
Using the GAA architecture

- We have designed a version of GAA (which we call **EMV-GAA**) which enables the existing EMV infrastructure to be used to provide generic security services in a simple and uniform way.
- It supports the same generic GAA interface as UMTS-GAA.

Roles

- The following roles are involved in the scheme.
  - *C*, EMV credit/debit card;
  - *T*, user terminal with card-reader;
  - *P*, user platform (T and C combined);
  - *I*, card issuing bank;
  - *B*, BSF server (online) with secure link to *I*,
  - *S*, GAA-aware application server (not involved in bootstrapping)

EMV-GAA



EMV-GAA bootstrapping I

- Involves *P* (user terminal and card), *I* (card issuer) and *B* (bootstrap server).
- Sets up authenticated secret master key (*MK*) between *P* and *B*, assisted by *I*.
  1. After receiving request, *B* generates R_B and sends it to *T*.
  2. User puts card *C* in reader, and *T* issues a **Generate AC** command to *C*, with *UN (Unpredictable Number)* set to R_B, other data *M*, and *Amount Authorised* set to zero.

EMV-GAA bootstrapping II

3. *C* returns an AAC, a 64-bit MAC computed using a secret key known only to *C* and *I*.
4. *T* generates R_T, and uses AAC as secret key to derive RES=$f_{\mathrm{AAC}}(R_T,R_B,\mathrm{Id}_B,M)$, where *f* could be HMAC.
5. *P* sends PAN (card number), R_T, *M* and RES to *B*, which forwards PAN and *M* to card issuer *I* (via a secure channel).
6. *I* recomputes AAC using received data, and sends it back to *B*.

Royal Holloway University of London
Information Security Group

EMV-GAA bootstrapping III

7. *B* uses the received AAC to recompute RES and compare it with the value received earlier (to complete authentication of *P*).
8. *B* generates master key as MK=KDF(AAC,R_T,R_B).
9. *B* computes XRES=$f_{\mathrm{AAC}}(R_B,R_T,\mathrm{PAN})$ and sends it to *P*.
10. *T* recomputes XRES and compares it with the received value to complete mutual authentication.
11. Finally *T* computes *MK*, and bootstrapping is complete.

- Only gives 64 bits of key entropy, but can generate two AACs to get greater security.

EMV-GAA use of bootstrapped key

- This is exactly the same as in UMTS-GAA (and generic GAA).

EMV-GAA properties

- Two major issues.
  - *Involves inserting an EMV card into a non-bank terminal – a risk in itself*. This can be resolved by requiring the bootstrap server to equip the user with a special card reader, as happens today with CAP (chip authentication program).
  - *The PAN is sensitive, and must be sent to the bootstrap server B*. This can be avoided using a one-off registration procedure.

GAA as a general framework

- GAA was originally designed to provide a way of exploiting the mobile phone security infrastructure.
- We have shown how it can be used to build on the EMV infrastructure.
- Could also be used as a framework for providing general purpose security services building on other pre-existing security infrastructures.

EMV-GAA – further developments

- Some EMV cards (supporting CDA or DDA as opposed to the widely used SDA) possess an RSA key pair and a certificate chain for the public key.
- Such a card can be requested to compute a signature by any card reader.
- This could be used to support GAA in a different way.
- It could also function as the basis of something like a universal PKI.

TC-GAA

- A further existing security infrastructure which could be used as the basis of a GAA service is the trusted computing infrastructure.
- The use of trusted computing (i.e. the TPM) to support GAA has been described in a previous paper (published in the proceedings of INRUST 2011).

Contents

- Security infrastructures
- GAA
- UMTS-GAA
- EMV-GAA
- Applying GAA variants
- Conclusions

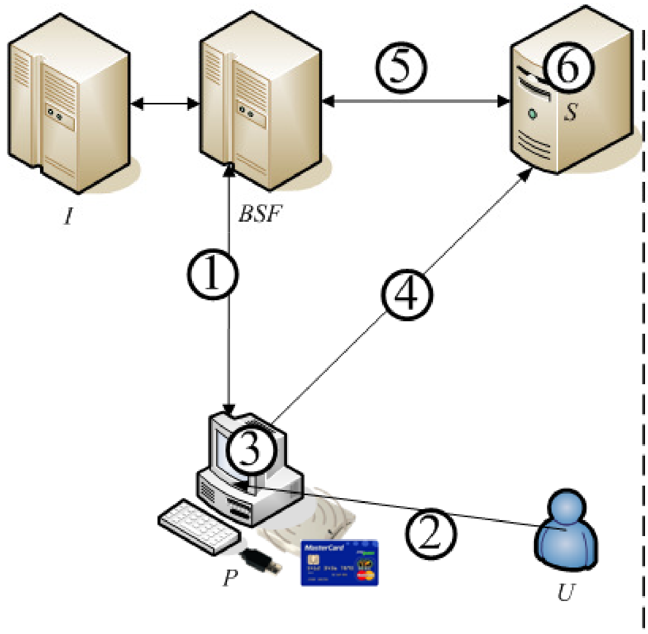
GAA-based one-time passwords I

- We consider one possible application of EMV-GAA, namely to enable the simple derivation of one-time passwords (OTPs).
- These passwords are based on a (potentially weak) long-term user password.
- The EMV-GAA session key provides protection against brute force password searches.

GAA-based one-time passwords II

- The OTP is computed as a function of the long-term user password and the short term application-specific session key.
- Compromise of the OTP does not enable a brute-force search for the password without knowledge of the session key.
- The EMV card used in the protocol does not need to be registered to the user – only needs to be trusted not to compromise the password.

EMV-GAA-OTP



1. $P \leftrightarrow BSF(I)$: bootstrap GAA credentials.
2. $U \rightarrow P$: *username* and *pw*.
3. P: derives a session key SK and computes $otp = f(SK, pw)$.
4. $U(P) \rightarrow S$: *B-TID*, *username* and *otp*.
5. $S \leftrightarrow BSF$: S fecthes SK and its lifetime.
6. S : checks whether or not SK is valid; if so, S recomputes *otp* for authentication; if not, S discards the request.

GAA OTP – other instantiations

- The notion of using a GAA session key to help generate an OTP from a long-term weak password applies to all instantiations of GAA.
- Indeed, in parallel work we have designed a series of simple OTP schemes using a GAA-enabled mobile phone.

GAA-based SSO

- We are also developing ways in which GAA could be used to build more general identity management solutions, including single sign-on schemes.
- Some work along these lines has already been standardised for UMTS-GAA, notably interoperation with CardSpace, OpenID and Liberty.

Contents

Trust

- All these GAA-based schemes require some level of trust in the TTP providing the BSF functionality.
- The exact degree of trust depends on the application.
- This may be a problem for some applications, but not for others, particularly for corporate environments.
- In any case, we all depend on TTPs for a variety of aspects of daily life (including banking, telephony, shopping, …).

Questions …