Exploiting existing security infrastructures

Chris Mitchell

Information Security Group
Royal Holloway, University of London
http://www.chrismitchell.net

Acknowledgements

- This is joint work with Chunhua Chen and Shaohua Tang (South China University of Technology).
- Chunhua Chen was a visitor at RHUL between 2009 and 2011.

Contents

- Security infrastructures
- GAA
- UMTS-GAA
- EMV-GAA and TC-GAA
- Applying GAA variants
- Privacy issues
- 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: (a) generate key pairs, and (b) obtain public key certificates from a CA.

Cost implications

- Setting up a new security infrastructure is a potentially very costly business.
- For example, distributing SIMs to all 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 ...

Examples of re-use I: CAP

- The Chip Authentication Programme (CAP) involves re-use of EMV cards for user-bank authentication.
- Users issued with special card readers.

Examples of re-use II: SIM apps

- The mobile phone (U)SIM can be used as a secure location for other applications.
- The SIM Application Toolkit (SAT) allows applications in the SIM to initiate actions (old technology).
- More recently, phone-based NFC payment card emulation using the SIM as a secure environment has been demonstrated.

Examples of re-use III: OAuth

- Facebook Connect implements the OAuth 2.0 standard, and uses it to provide a single sign-on (SSO) service.
- This builds on the relationship Facebook has with its clients.
- Facebook Connect allows users to sign-on to applications (e.g. Facebook-affiliated websites) using their Facebook account, and also enables such applications to access Facebook-hosted user data, subject to user authorisation.

Issues

- Such re-use of existing trust relationships has clear privacy and security issues.
- We note the following privacy issues:
  - existing infrastructure owner will learn about user interactions with other entities;
  - infrastructure owner could use information to build user profile, e.g. for focused advertising;
  - other entities will learn about user's existing trust relationships.

Scope of this talk

- Look at a specific general purpose architecture for infrastructure use.
- Consider instantiations of this architecture.
- Consider applications of this architecture.
- Look at privacy and security issues.

Contents

- Security infrastructures
- <u>GAA</u>
- UMTS-GAA
- EMV-GAA and TC-GAA
- Applying GAA variants
- Privacy issues
- Conclusions

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.

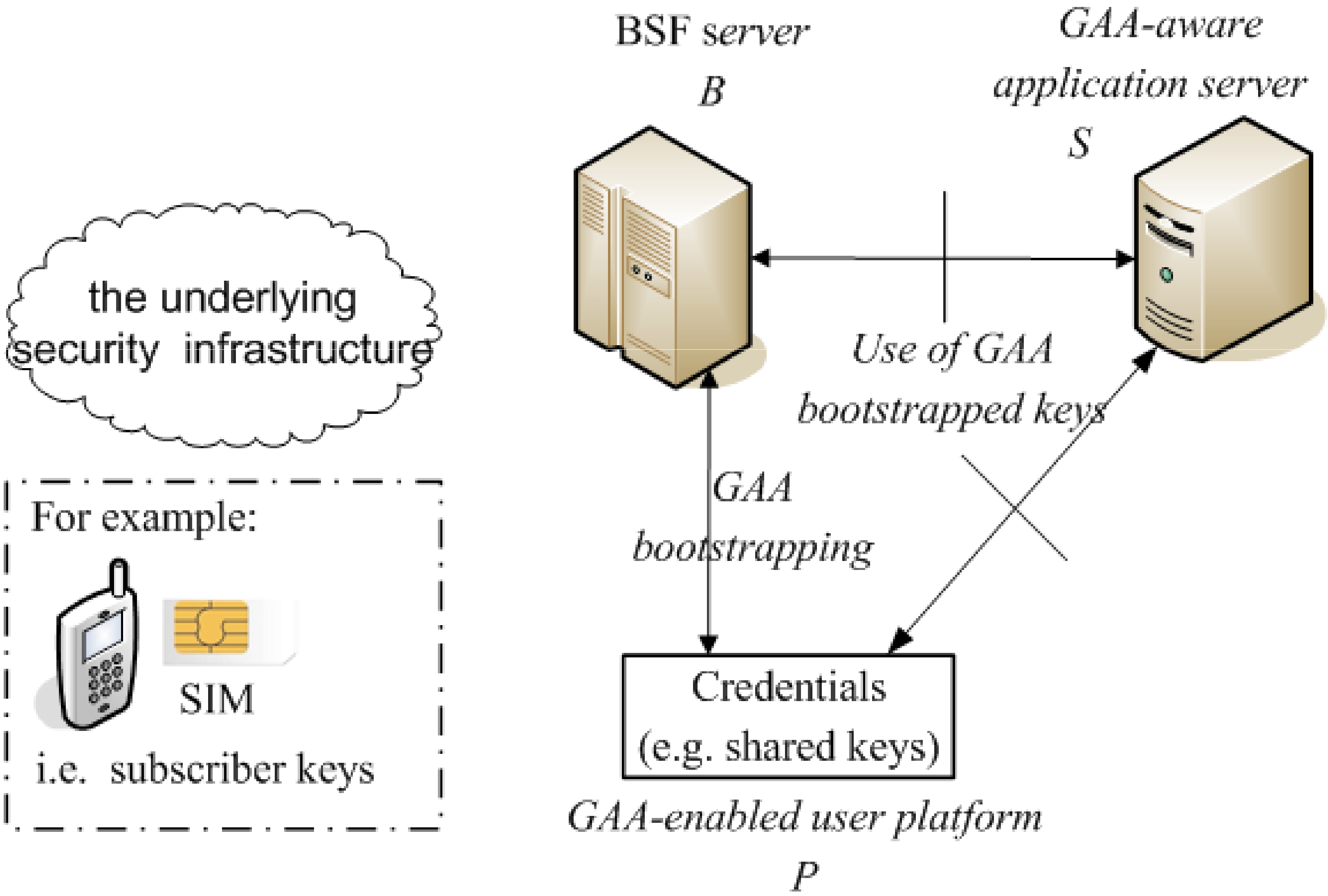
GAA roles

- 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 a GAA-enabled user platform and a GAA-aware application server.
- User platform must have an existing security context with a party working with the GAA service provider (BSF).
- The target server must have a relationship with the GAA service provider (BSF).

GAA overview



GAA procedures

- Two main procedures:
  - **GAA bootstrapping** – Establishes a secret master key *MK* (+ a key identifier *B-TID* 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

- Security infrastructures
- GAA
- <u>UMTS-GAA</u>
- EMV-GAA and TC-GAA
- Applying GAA variants
- Privacy issues
- Conclusions

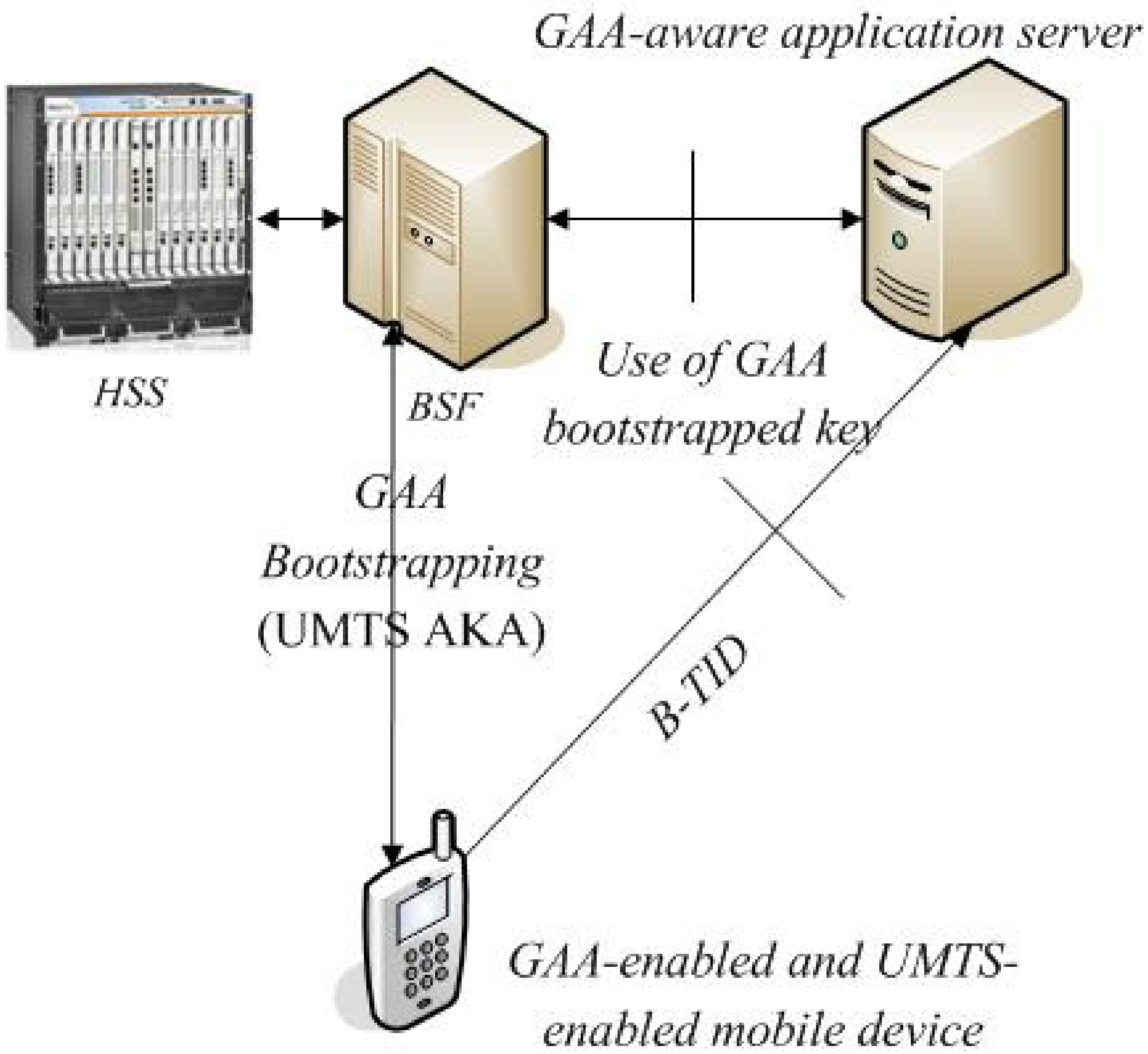
UMTS – background

- 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; HSS is 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;
  - USIM shares key with HSS of issuing network;
  - BSF connects to the HSS for the USIM (BSF could be owned by same operator);
  - UMTS Authentication and Key Agreement protocol (UMTS AKA) used to establish *MK* between GAA-enabled user platform and BSF (*MK* is concatenation of *IK* and *CK*).

UMTS-GAA


GAA-aware application server
HSS
BSF
Use of GAA
bootstrapped key
GAA
Bootstrapping
(UMTS AKA)
B-TID
GAA-enabled and UMTS-
enabled mobile device

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

- Security infrastructures
- GAA
- UMTS-GAA
- <u>EMV-GAA and TC-GAA</u>
- Applying GAA variants
- Privacy issues
- Conclusions

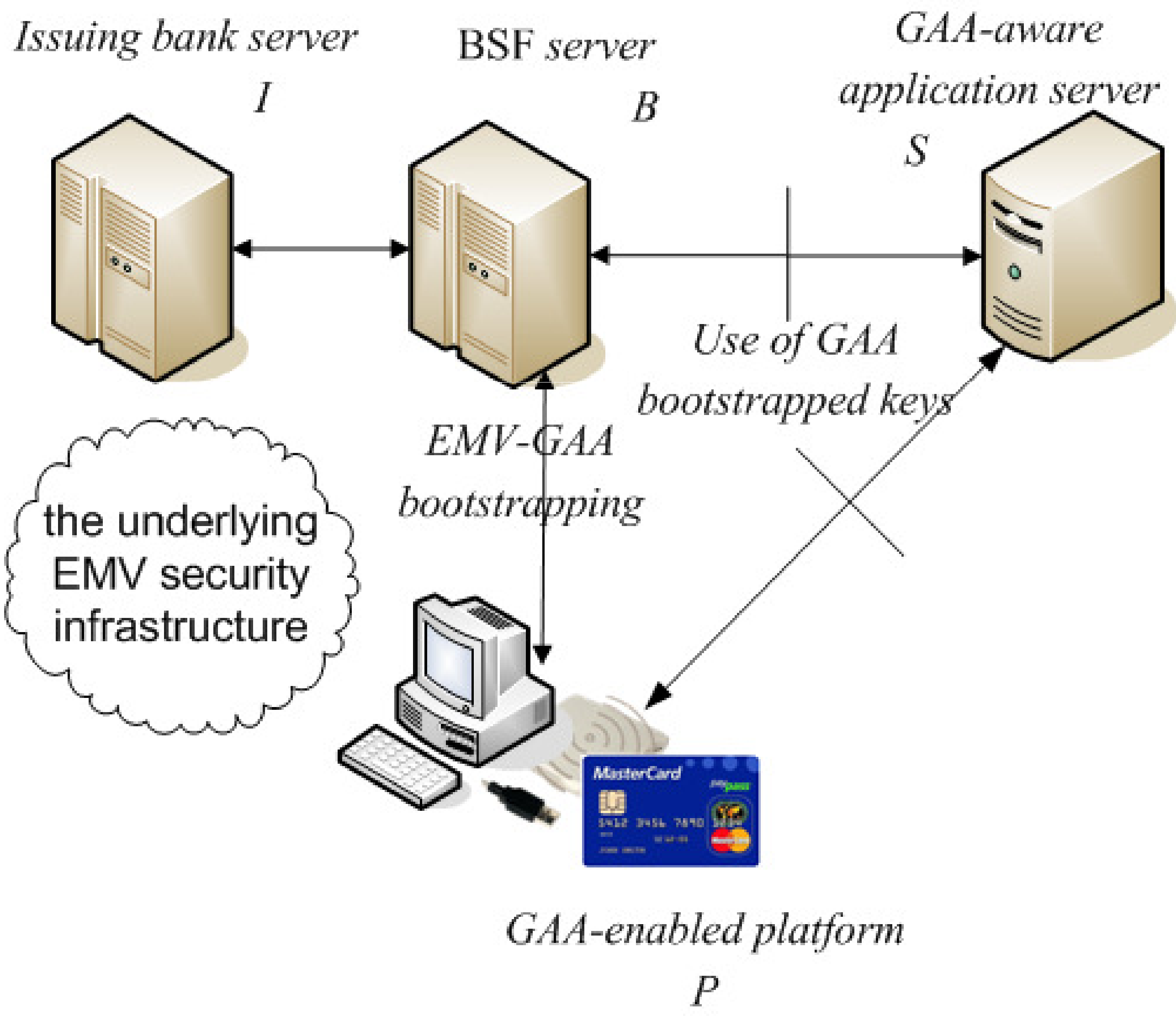
Using the GAA architecture

- Chen et al. (2012) have designed a version of GAA (**EMV-GAA**) which enables 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.
- [For further details see SecureComm 2012 proceedings].

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


Issuing bank server
I
BSF server
B
GAA-aware
application server
S
Use of GAA
bootstrapped keys
EMV-GAA
bootstrapping
the underlying
EMV security
infrastructure
MasterCard
GAA-enabled platform
P

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*.

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_{AAC}(R_B,R_T,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.

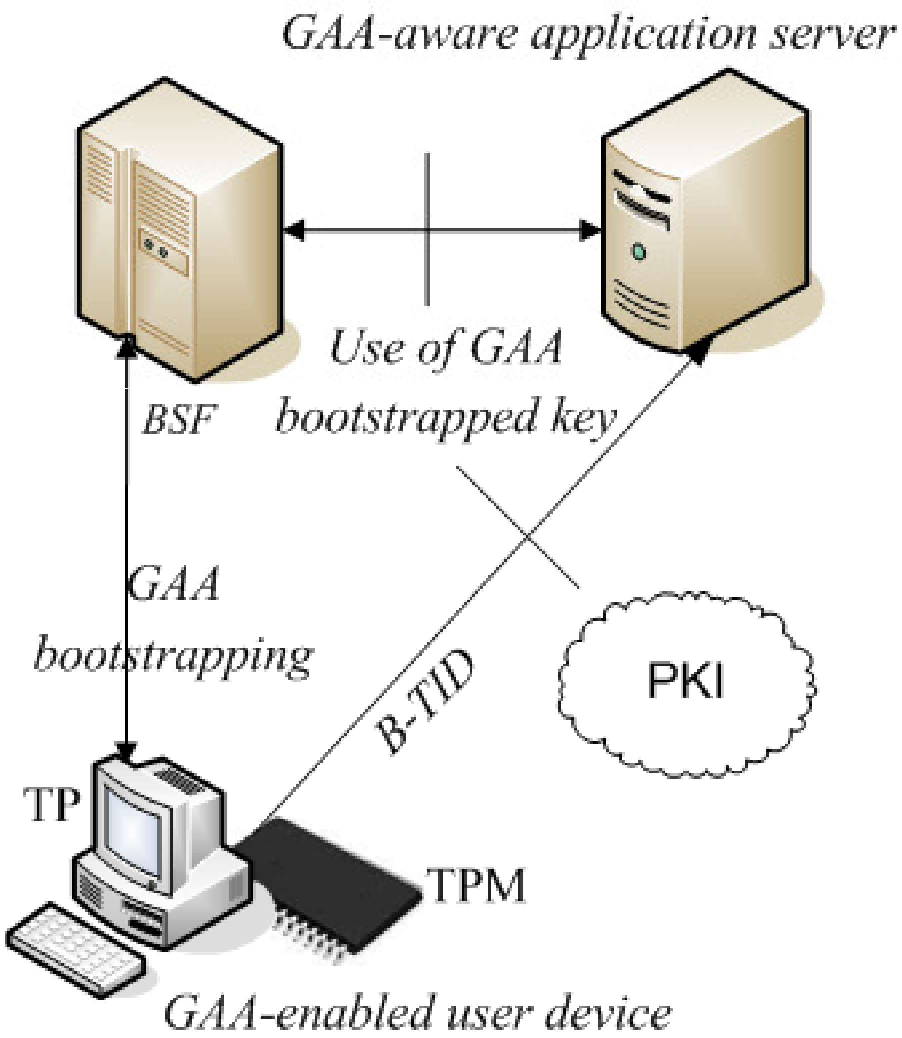
EMV-GAA – further developments

- 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 (**TC-GAA**) is the trusted computing infrastructure.
- The use of trusted computing (i.e. the TPM) to support GAA has been described by Chen et al.
- [For further details see the proceedings of INTRUST 2011].

TC-GAA – overview



TC-GAA – a sketch

- The TPM on the client machine is instructed to generate a new encryption key pair.
- The public key is then signed (certified) by the TPM using a previously generated Attestation Identity Key (AIK).
- The newly generated certificate is now sent to the BSF along with a previously generated Privacy-CA-generated certificate for the AIK public key.
- After verifying the two certificates, the BSF generates an *MK*, encrypts it using the TPM-generated public key, and ships it back to the TPM.
- This complete the TC-GAA bootstrapping procedure.

TC-GAA properties

- Note that the derivation of *SK* can be very similar to the generic case.
- It is interesting to observe that, unlike UMTS-GAA and EMV-GAA, the 'issuer' of the TPM is not actively involved.
- Any TTP can function as the BSF without a trust relationship with a further third party.
- This enhances the privacy properties.
- This advantage results from building GAA on asymmetric crypto rather than shared secrets.

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 and TC infrastructures.
- Could also be used as a framework for providing general purpose security services building on other pre-existing security infrastructures.

Using infrastructures directly

- It is perfectly possible to design applications building directly on the trusted computing infrastructure.
- Substantial literature now exists.
- However, secure application protocols are non-trivial to design.
- Trust relationships can be very unclear.
- **Infrastructure provider will have access to all session keys.**

Contents

- Security infrastructures
- GAA
- UMTS-GAA
- EMV-GAA and TC-GAA
- <u>Applying GAA variants</u>
- Privacy issues
- Conclusions

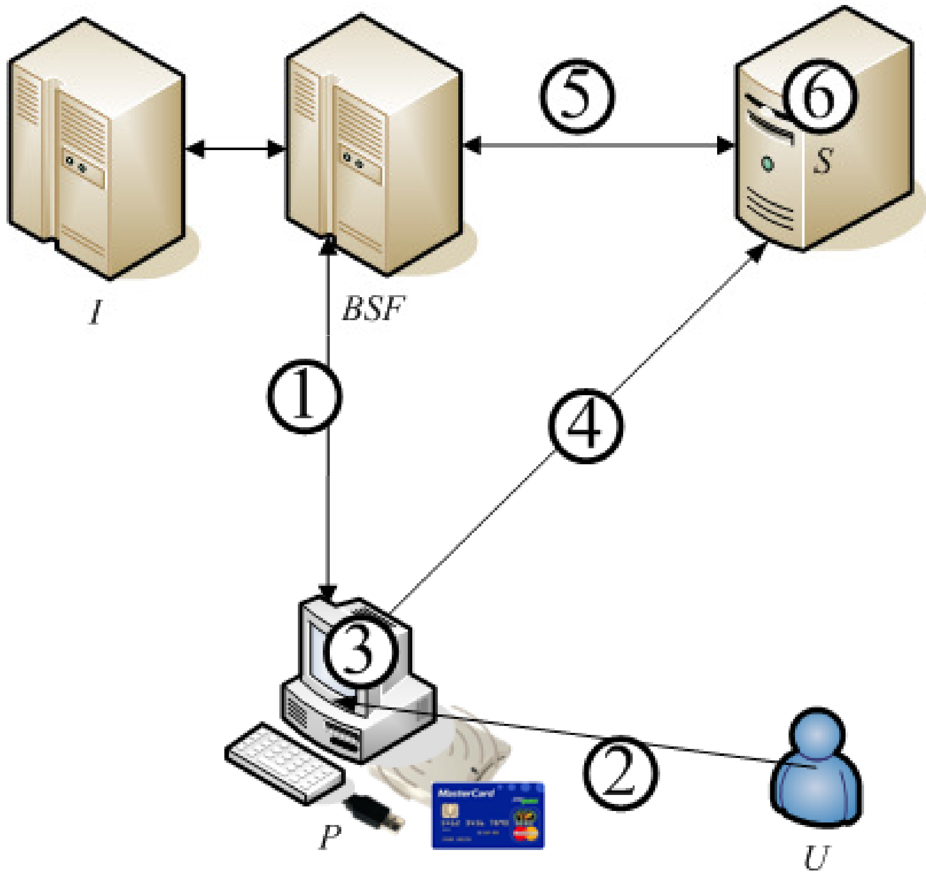
GAA-based one-time passwords I

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

GAA-based one-time passwords II

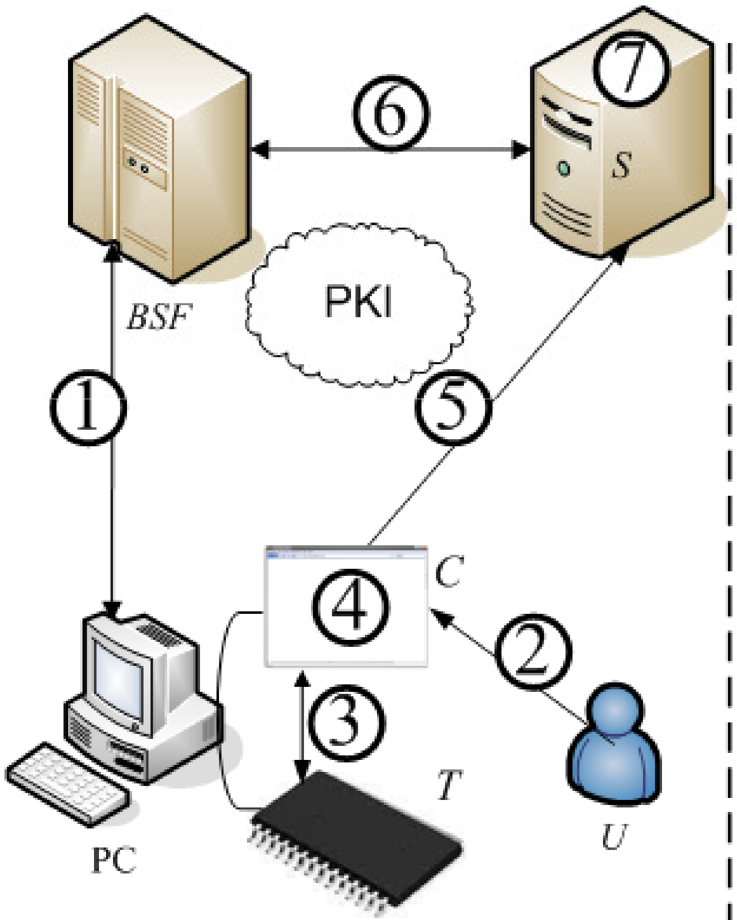
- The OTP is computed as a function of the long-term user password and the short term application-specific session key.
- Compromise of the OTP does not enable a brute-force search for the password without knowledge of the session key.
- If EMV-GAA is used, the EMV card used in the protocol does not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

EMV-GAA-OTP



1. *P* ↔ *BSF*(*I*): bootstrap GAA credentials.
2. *U* → *P*: *username* and *pw*.
3. *P*: derives a session key *SK and* computes $otp = f(SK, pw)$.
4. *U*(*P*) → *S*: *B-TID*, *username* and *otp*.
5. *S* ↔ *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.

TC-GAA-OTP



1 $T \leftrightarrow BSF$: bootstrap GAA credentials.

2 $U \rightarrow C$: *username* and *pw*.

3 $C \leftrightarrow T$: interact for a session key *SK*.

4 *C*: computes $otp = f(SK, pw)$.

5 $U(C) \rightarrow S$: *B-TID, username* and *otp*.

6 $S \leftrightarrow BSF$: *S* fecthes *SK* and its lifetime.

7 *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

- Security infrastructures
- GAA
- UMTS-GAA
- EMV-GAA and TC-GAA
- Applying GAA variants
- Privacy issues
- Conclusions

Privacy

- We can think about privacy in terms of who gains access to what PII of a user.
- Indeed, under some definitions, privacy equates to PII management.
- Regardless of whether this is true, the generation, control, dissemination and use of PII is certainly a core part of privacy.
- Look at privacy aspects of infrastructure re-use, using GAA as an example.

Trust in Infrastructure provider

- In GAA using symmetric crypto, the infrastructure provider is asked for an *MK* by the BSF during every bootstrapping.
- The *MK* is not specific to any server.
- It gives general information about user activity.
- A ‘nosy’ infrastructure provider could use the MK to compute server/application-specific keys if it eavesdropped on connections.
- **Risk**: depends on nature of provider – e.g., telecommunications providers and ISPs can already eavesdrop ...

Trust in proxy (BSF in GAA)

- The BSF learns a lot:
  - it knows identifies of all servers and applications a user interacts with;
  - it has access to all the secret keys established with these servers.
- BSF could be a ‘special purpose’ TTP.
- **Risk**: great – a high level of trust is needed in the BSF.

Trust in Service Provider

- Service provider will learn which proxy (BSF) user prefers.
- However, the *SK* given to the server should yield no useful information about the *MK* or any other Service Providers in use.
- **Risk**: seems relatively low.

Impact of cryptography

- Asymmetric cryptography has certain privacy advantages over symmetric cryptography, at least in context of GAA.
- If underlying infrastructure involves asymmetric key pairs, then there is no need for infrastructure provider to be actively involved in bootstrapping.
- This reduces privacy threat from infrastructure provider.

Improving privacy

- How can we improve privacy?
- Major threat is that BSF knows all keys.
- User and server could use secret *SK* as basis for an authenticated Diffie-Hellman key exchange.
- Newly established key would not be available to the BSF.

Contents

- Security infrastructures
- GAA
- UMTS-GAA
- EMV-GAA and TC-GAA
- Applying GAA variants
- Privacy issues
- Conclusions

Conflicts

- As has been observed many times, there is a major conflict between:
  - the (claimed) desire of users for privacy,

  versus:
  - the observable large scale use of services which potentially compromise privacy, e.g. Facebook itself and Facebook Connect.
- Re-using an existing security infrastructure involves some privacy compromises, but it may be very convenient.

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.
- We can also take extra precautions, e.g. by using multiple BSFs.

Questions …