Mobile Security

Chris Mitchell
http://www.isg.rhul.ac.uk/~cjm

ROYAL HOLLOWAY, UNIVERSITY OF LONDON
Information Security Group

This talk …

- Briefly introduces security terminology.
- Discusses a range of security problems arising in a pervasive and mobile computing environment.
- Also covers possible directions for solutions to some of the problems.
- Although it covers number of topics, they are all inter-related (as will become obvious!).

Environment

- Ubiquitous computing environment …
- For this talk this is assumed to mean an environment in which multiple devices, some personal, some mobile, combine to provide an all-pervasive computing and communications service to end-users.
- Requires automatic configuration of certain aspects of some devices, since it is assumed that there is no global management infrastructure.

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

What is security?

- Security is a difficult notion to define precisely.
- In discussing information systems, security is often equated to CIA: *Confidentiality*, *Integrity* and *Availability*.
- This division is now recognised to miss some important aspects of security, e.g. including *Accountability*.
- Security typically defined in terms of preventing certain threats.

Communications security services

- In a communications context, security often considered in terms of *services*, including:
  - *Authentication* (including *origin authentication* and *entity authentication*),
  - *Access control* (where *Authorisation* for access to resources is required).
  - *Confidentiality*,
  - *Integrity* (which typically involves *detecting* changes rather than *preventing* them), and
  - *Non-repudiation*.

Terminology (not necessarily universally agreed!)

- *Identification*: Learning a (claimed) identifier for an entity – may be a pseudonym.
- *Authentication*: Verifying that an entity does correctly possess a certain identifier.
- *Authorisation*: Checking that a particular entity has permission to access certain resources. **Authorisation requires authentication since permissions typically bound to an identifier.**

Security requirements in mobile world I: *Identification/authentication*

- Mobile and fixed devices need to identify and authenticate one another.
- This needs to work in cases where there is no pre-existing relationship between the two devices.
- Techniques used must, where necessary, protect privacy of device owners, e.g. provide anonymity, prevent linking of transactions, …

Security requirements II: *Confidentiality*

- Communications between mobile and personal devices may involve the transfer of sensitive data.
- Stored data may also need protection.
- Other information, such as identifiers, may need protection against disclosure (e.g. to support privacy).

Security requirements III: *Availability*

- Communications channels and individual devices need to be available for use when required.
- This requires some measure of defence against Denial-of-Service (DoS) attacks.
- DoS attacks may involve jamming communications, using all computing or storage resources, using all stored power, …

Security requirements IV: *Integrity*

- Important data will often need to be protected against unauthorised change, including re-ordering, duplication, etc., when stored or transmitted.
- Integrity protection for transmitted data closely related to data *origin authentication*.
- Data here includes executable objects (code), operating systems, …

Security requirements V: *Non-repudiation*

- It is sometimes necessary to protect against one entity denying having taken a particular action, e.g. sending or receiving a message.
- Applications include making binding contracts, e.g. for buying/selling goods, and committing to payments.
- Non-repudiation services exist to provide this kind of security protection.

Security requirements VI: *Authorisation*

- The controller of a resource (e.g. stored data, processing capability, communications capability, etc.) will typically need to know that a user requesting access to a resource is *authorised* to have this access (to exert access control).
- This covers such issues as whether or not a mobile agent should be permitted to run.

Security mechanisms

- There are a large variety of techniques that can be used to provide/enforce security services.
- They include:
  - Cryptographic mechanisms,
  - Security protocols,
  - Trusted Third Party (TTP) functions,
  - Physical security,
  - Design security.

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

On privacy

- It is important to distinguish between security and privacy.
- Privacy is not just a special case of security – there are interesting interactions between security and privacy.
- It is important to appreciate that security and privacy are very different notions – indeed the two sometimes conflict.

Privacy background

- Many of the devices providing the ubiquitous environment will be personal devices; they may thus 'leak' personal information to each other.
- In a context where devices cannot be assumed to belong to a single trusted domain, there are thus major privacy issues.

Interrogation of mobile devices

- Communications protocols for mobile devices inevitably require some form of routine 'polling'.
- Responses to polls (e.g. from a network access point) need to contains some kind of identifier, e.g. a network address.
- Thus can be used to 'track' devices, and potentially track the location of device owners.
- Solutions? GSM/3GPP use temporary identifiers (pseudonyms) distributed in a way that prevents linking. Provide confidentiality protection for exchange.

Use and abuse of authentication

- Authentication of a device can pose a denial of service threat.
- For example, if protocol requires one device to store state and/or do computations, repeated fake requests can cause memory/processing exhaustion.
- Solutions? Use stateless protocols. Require requester to do at least as much work as the responder.

Location information use/privacy

- Service providers in a ubiquitous computing environments may wish to provide services based on user location, e.g. targeted advertising, emergency services, broadcast blackout, …
- Owner of computing device may wish to restrict dissemination of such location info.
- How should this be controlled?
- Solutions? Anonymity. Mandatory inclusion of policy data with location information. TTPs.

Denial of Service versus privacy

- In any protocol it seems that one party has to reveal their identity first. This argues that (mostly) the requester of service should reveal their ID last. (P2P an exception?)
- However, this potentially increases the risk of Denial of Service attacks against the responder.
- Indeed, more generally, the tension between DoS-resistance and user privacy has been noted by a number of authors.
- Solutions? New ideas needed?

ROYAL HOLLOWAY, UNIVERSITY OF LONDON
Information Security Group

Accountability versus privacy

- Anonymity to protect privacy may cause major problems in making users accountable for their actions.
- An audit trail (present to provide accountability) is useless if the real owner of a pseudonym cannot be determined.
- Must try to balance the need for privacy and the need for accountability.

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

ROYAL HOLLOWAY, UNIVERSITY OF LONDON
Information Security Group

General problem

- For the purposes of this lecture, an ad hoc network is a collection of communicating devices with no pre-existing relationships or infrastructure.
- Many security issues arise in establishing working relationships in such a network, e.g.:
  - Initial trust setting;
  - Managing collaborative activities (e.g. routing);
  - Authentication, authorisation, …

ZeroConf Requirements

- The need to establish relationships between mobile devices dynamically has given rise to work on *Zero Configuration* (ZeroConf).
- To reduce network configuration to zero (or near zero) in Internet Protocol (IP) networks, it is necessary to:
  - Distribute IP addresses (without a DHCP server),
  - Distribute multicast IP addresses, if necessary (without a multicast server),
  - Provide name resolution (without a DNS server),
  - Find and list services (without a directory service).

  [See: M. Hattig, *ZeroConf Requirements*, Internet draft, Zeroconf WG].

ROYAL HOLLOWAY, UNIVERSITY OF LONDON
Information Security Group

Requirements for *Secure* ZeroConf

- To distribute IP addresses securely (without a DHCP server),
  - Preventing address hoarding/duplication by rogue nodes (may use IPv6 or trusted computing technology).
- To distribute multicast IP addresses securely, if necessary (without a multicast server).
- To provide name resolution securely (without a DNS server),
- To securely find and list services (without a directory service),
  - Provide for host and entity authentication.
- Key Establishment & Exchange (without a CA or PKI infrastructure).
  - Possibly using an ID based PKI.

Secure ZeroConf process

- Identification
  - Obtain an unique identifier: Detect Address Duplication.
- Registration
  - Joining the network;
  - Discovering neighbours;
  - Advertise oneself.
- Service Discovery and Advertisement.
  - Service Registration and Deregistration;
  - Secure Discovery;
  - Secure Delivery;
  - Availability.
- Secure Communication.

Trust establishment

- One fundamental issue for two devices in an ad hoc network (with no pre-existing relationship) is deciding whether to trust one another (and by how much).
- What resources or services should one node make available to another?
- Can another node be trusted to provide a communications service without eavesdropping, manipulating messages, and/or selectively dropping packets?

Reputation schemes

- One solution is to try to dynamically ‘measure’ the trustworthiness of another node.
- Each node maintains an assessment (typically a numerical score) of the trustworthiness of its neighbour nodes.
- This would typically be derived by monitoring the behaviour of the node, possibly combined with assessments passed on by other nodes.
- Such schemes are widely used, e.g. on eBay.
- However, such schemes are also easily spoofed.
- Many schemes have been proposed, but the robustness of schemes against deliberate attack has rarely been assessed.

Currency schemes

- Another solution proposed in some scenarios, is to use a virtual ‘currency’ to reward nodes performing service in an ad hoc network.
- Currency needs to be unforgeable!
- Other problems arise – e.g.
  - shortages of currency can cause major inefficiencies in the network;
  - how to start the scheme started, i.e. how does the currency get allocated initially?

Need for stable identities

- Any reputation scheme requires some means to assign identifiers to nodes in a way that prevents a bad node changing its identity, e.g.
  - to escape a bad reputation;
  - to steal the identity of a node with a good reputation.
- This is the ‘stable identifiers’ problem.
- The identifier could be pseudonym – that is, in many cases anonymity can be preserved.

Automatic address assignment

- In an ad hoc network, newly admitted devices will typically need to be assigned a network address (or addresses), e.g. an IP address.
- In the absence of a fixed infrastructure this is problematic.
- Solutions can easily lead to the possibility of denial of service attacks (e.g. where a newly admitted node cannot get an address).

Threats to address assignment

- Precise nature of threat depends on environment.
- Example threats include:
  - *Address squatting*: a malicious node prevents a newly arrived node from getting an address by repeatedly claiming that a selected address is already in use;
  - *Sybil attacks*: malicious nodes use many addresses to get unfair share of resources, or to avoid responsibility for actions (this is a major problem in p2p systems, which share many properties of ad hoc nets).

Possible solutions

- Use of *reputation systems* to prevent address squatting, but these require stable identities.
- Providing *stable identities* typically requires some kind of infrastructure.
- Solutions:
  - For devices with a SIM/USIM – build on pre-existing trust relationship with operator;
  - For TCG-compliant devices – use keys and certificates installed in the Trusted Platform Module (TPM);
  - Use cryptographically generated addresses (only a partial solution).

Cryptographically generated addresses

- Generate address by applying a one-way function to a public key from a key pair.
- Can prove ownership of address by giving public key and proving knowledge of corresponding private key.
- As long as address has enough bits, it will be infeasible for anyone to find a key pair which generate someone else's address.
- This prevents address theft, but does not prevent Sybil attack.

Routing in ad hoc networks

- Many protocols designed to enable distributed routing in ad hoc environment.
- Such routing protocols rely on cooperation between nodes (multi-hop operation).
- However, such schemes are prone to a variety of attacks, including 'selfish' behaviour.
- How to remove freeloaders?
  - Build behaviour monitoring into routing protocol, but still need stable identities …

Threat model for ad hoc networks

- External Threats:
  - Passive eavesdropping, and
  - Active interference.
- Internal Threats:
  - Failed nodes,
  - Badly Failed nodes,
  - Selfish nodes, and
  - Malicious nodes.

Possible malicious node attacks

- Denial of service, for example:
  - Exploit route maintenance,
  - Attack sequence number mechanisms.
- Misdirect traffic, for example:
  - Masquerade as another node and send control packets,
  - Reply to route requests falsely indicating better routes.
- DoS attacks can cause misdirection, and misdirection is also a form of DoS.

Selfish behaviour

- An ad hoc network is a finite resource,
- Cooperation is essential for communication to occur, but
- If a node does not forward packets for other nodes, then it will save power and bandwidth for its own packets,
- If every node behaves in this way, then the result is an ad hoc ‘collection’ of nodes rather than a network.

A definition of Selfishness

- Selfish nodes are those that agree to perform a service, but then refuse to provide that service when requested, in order to save power and enhance their own quality of service.
- Nodes which consume services for their own gain, and in doing so prevent others from using the same services, exhibit a different form of selfishness which is defined as greediness.

Non-forwarding behaviour

- Non-forwarding behaviour can be caused by:
  - Failed communications,
  - Failed nodes,
  - Sleeping nodes,
  - Selfish nodes.
- Any mechanisms dealing with selfish nodes must take all these possibilities into account.
- They also need to manage the apparent contradiction between the security requirement of availability and the goal of energy-conservation.

The problem with Selfishness I

- Easy to hide information:
  - High throughput routes are not even advertised.
- Dropping of control packets.
- Partial dropping of packets.
- Bandwidth, energy and resources are wasted.
- Therefore need to maintain reasonable throughput in the presence of both failed and selfish nodes.

The problem with Selfishness II

- MAC layer mechanisms:
  - Acknowledgements only indicate that a neighbour has received a packet, and not forwarded it;
  - Cannot rely on using promiscuous mode, as it is vulnerable to collisions and nodes moving in and out of each other's radio range.
- Difficult to distinguish between failed nodes and selfish nodes:
  - Are packets being dropped because of selfishness, a lack of energy, natural packet loss or congestion?

Dealing with Selfish Nodes

- Solutions in the lower layers:
  - Schemes integrated into routing protocols,
  - Reputation mechanisms, and
  - Currency models.
- Solutions in the upper layers:
  - Usage Policy:
    - Provide limited services,
    - Offer incentives, such as
    - Billing discounts.

Modified routing architecture

- One technique (due to Po-Wah Yau) involves monitoring node behaviour within a 2-hop radius, with assistance of nodes at 1-hop distance.
- Previously, most proposed methods only considered nodes at 1-hop distance – this, however, does not enable a reliable assessment of all relevant local behaviour.

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

Location of security functionality

- Where security functionality is located in a protocol stack can have a significant effect on security provision, including:
  - Encrypting at the application layer will not hide any of the lower layer addresses and routing information. May also cause problems for firewalls.
  - Integrity protection at individual link level will not provide end-to-end integrity protection.

End-to-end versus point-to-point

- Need for security between service provider and service consumer argues for end-to-end authentication.
- Need for control of access to resources, e.g. network access, argues for point-to-point authentication.
- If both provided in 'unlinked' way then man-in-the-middle attacks can become possible.
- Great care needed in combining protocols at different levels in protocol hierarchy.

Protocol statefulness

- As mentioned previously, protocol state can be used as a means of launching DoS attacks.
- 'Accepted wisdom' is to require protocols to be stateless, at least for responder.
- However, there is an efficiency cost (state must be shipped in protocol messages). It also either requires synchronised clocks or regular key changes (a bit like state).

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

Background

- Use of crypto requires either shared secret keys (using symmetric crypto) or trusted copies of public keys (using asymmetric crypto).
- Shared secrets can be set up via a mutually trusted TTP.
- Public keys can be obtained via public key certificates as part of a PKI, although trusted means to verify certificates (CA public keys) required.

Heterogeneous networks

- The pair of devices may not share an online TTP (or even share 'trust-connected' TTPs).
- Public key crypto (and PKI) looks more promising, but it is still necessary to have mutually verifiable certificates. Finding certification paths (where one CA certifies the public key of another CA) could be infeasibly complex for a bandwidth-limited device.
- Solutions? Delegated Path Discovery/Delegated Path Validation.

PKI interoperability

- Finding a certification path is by no means only problem with using PKI.
- Certificates issued by different CAs (with different policies) may 'mean' different things – e.g. different liability protection, different ID checking for certificate issue, etc.
- Certificate status management systems may vary.

ID-based cryptography

- One possible solution to key management problems is used of ID-based crypto.
- Here a user public key is derivable from a user identifier (possibly plus other data).
- Requires TTP to issue private keys (and TTP public parameters to derive public key from ID).
- Hence we have major interoperation issues if two devices served by different TTPs.

Exploiting the mobile operator relationship

- One possible solution to the key management (and authentication) problem is to exploit existing relationships.
- One such relationship is that between end users and mobile phone network operators.

What is there to exploit?

- Many problems in future pervasive computing environment arise from lack of security infrastructure.
- The GSM SIM and UMTS USIM, present in all mobile phones, represent an existing security infrastructure (based on shared secret keys).
- This could potentially be exploited to provide new security services, over and above existing purpose (mobile device/network authentication and key establishment).

Single Sign-On (SSO) support

- The SIM/USIM could be used as the basis of a service which enables the identity of a mobile device to be verified by a network service provider (SP).
- The network operator would use the SIM (in some way) to verify the mobile device identity, and would then vouch for the identity to the SP.

Secure payment support

- Here, one major problem is to verify the identity of the mobile device owner to a financial institution (in some ways just a special case of SSO).
- However, can help if authentication process tied to both the payment (value) and the goods/services being provided.

3GPP GAA

- Value of these ideas recognised by 3GPP which has published its *Generic Authentication Architecture* (*GAA*).
- This enables the mobile network operator to offer its authentication function as a service.

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

Background

- Desire for an Internet single sign-on solution.
- That is, instead of a user authenticating him/herself to multiple service providers (SPs), the user authenticates him/herself to an Identity Provider who then provides assurances (*assertions*) regarding the user identity to SPs.
- This requirement becomes even more important in a ubiquitous environment, where a user will not wish to authenticate him/herself to every device/service.

SSO terminology

- When a user wishes to make use of a service, the service will typically wish to be sure who the user is (e.g. for charging purposes).
- This requires the user to provide an *identity*, and also to give the means (via one or more *credentials*) for the service provider to *authenticate* the claimed identity.

Identities

- A user may have many identities (with associated identifiers) for use with different service providers.
- For example:
  - an employee may have an employee number for use with his/her employer;
  - a citizen has one or more numbers for interactions with government;
  - a user of Internet services (e.g. messaging) may have multiple names, each used with a set of service providers.

Credentials

- To enable a service provider to authenticate a user as a legitimate holder of an identity, the user may be required to provide one or more credentials.
- Possible credentials include:
  - a password;
  - a biometric sample;
  - a public key certificate;
  - a MAC computed using a shared secret key;
  - a signature on a challenge provided by the service provider.

Authorisation

- Once an entity has been authenticated, the service provider needs to decide whether or not to grant the requested service.
- This is referred to as *authorisation* (i.e. is the holder of this identity authorised to access this service?).
- This could, for example, be supported using server-held Access Control Lists (ACLs).

Microsoft Passport

- (Originally) a proprietary SSO solution, which also (originally) involved the possibility of managing other personal data, all stored on a server somewhere ...
- Problems with guardians of end-user privacy, including European Commission.
- Passport succeeded as user authentication method for Yahoo.
- Passport failed as a SSO solution.

Lessons from Passport

- Microsoft's experience with Passport has been rather painful.
- They tried to solve the problem of identity management by becoming ***the*** global identity provider.
- This idea failed abysmally – the main lesson is that there will never be such a global identity provider.
- This has led to InfoCard, as a means of supporting an identity ecosystem with multiple providers …

Liberty Alliance

- Consortium set up to provide an open system (protocol suite) to support SSO.
- Provides variety of alternative means of transferring assertions from IP to SP.
- E.g. using SOAP, web redirection.
- Possible problems, as with any scheme using web redirection, if man-in-the-middle attacks.

WS Federation

- Part of Web Services Security.
- Covers federation of identifiers, and also allowed ‘brokering’ of identity/authentication services.
- Can be used as the basis of an SSO scheme.

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

What is trusted computing?

- Computer security has a long history, and many secure computer systems have been produced and sold.
- Almost all of them depend on the assumption that the computer hardware will be physically secure, and managed by trusted personnel.
- Physical access to the machine will typically allow software integrity to be compromised

Multi-user systems

- Many systems (e.g. Unix, Windows 2K/XP) designed to allow users to protect their data and resources against other users of same machine.
- All based on access control systems.
- Again typically dependent on physical security of machine.

Computer security – external view

- If a (secure) computer digitally signs a message, then trust in messages depends on:
  - trust in computer software, and
  - trust in physical security of hardware (and in correct application of security procedures by administrators).
- Makes sense in conventional 'computer centre'.

PC security

- Perhaps an inherent contradiction!
- PCs are not stored in a physically secure environment.
- Even though modern versions of Windows (and Linux) have multi-user security features, users and programs often run as administrator.
- There are many ways that the operating system integrity can be damaged.

The need for trusted computing

- Today, neither the user of a PC nor a communicating party can trust very much at all about a PC.
- This is despite major efforts to improve security of Windows.
- Anyone with access to the PC hardware can modify Windows (e.g. by removing hard disk and changing files).

Trusting a PC – more bad news …

- Even if the user looks after the physical security of their PC, there are many other threats to system integrity.
- Modern operating systems and applications are highly complex and it is almost impossible to remove all vulnerabilities.
- Users can easily accidentally run malicious software which can damage system integrity.

Need for trust I

- User may want to trust the integrity of their PC.
- For example, the PC may be used for:
  - managing a bank account,
  - performing e-commerce transactions,
  - managing personal information,
  - ...

  all of which require *user* trust in the PC.

Need for trust II

- Third party may want to trust integrity of PC.
- This could be for a variety of reasons, e.g.:
  - 3rd party is a bank: PC being used for e-commerce,
  - 3rd party is a content provider: PC performing DRM,
  - PC performing other security functions (e.g. authentication, key management) on behalf of 3rd party,

  all of which require *third party* trust in the PC.

Role of Trusted Computing

- Enables trust in integrity of PC based on combination of software and hardware.
- Trusted Computing does not just apply to conventional PCs: equally relevant to PDAs, mobile phones, broadcast receivers, …

Trusted computing specifications

- Many parallel and related developments:
  - TCPA/TCG
  - Palladium/NGSCB
  - La Grande technology
  - Perseus, …
- All seek to provide a ‘trusted computing environment’ within a device, in which external parties can verify that interactions are taking place with a particular piece of code, and that data will only be available to this code.

TCG

- Trusted Computing in the sense of this talk dates back to late 1990s.
- Consortium of major manufacturers started TCPA (Trusted Computing Platform Alliance).
- This has morphed into TCG, the Trusted Computing Group.

Trusted computing in mobile devices

- The future use of trusted computing technology in Europe will (we hope) be supported by the current FP6 project: *Open Trusted Computing* (www.opentc.net).
- Project is developing a framework and open source software for use of trusted computing hardware on open platforms.
- Project involves 23 partners across Europe and started in late 2005.

Current position

- Operational technical working groups for:
  - Future TPM, trusted platform module
  - PC specific implementation specifications
  - New TSS, TCG software stack specifications, as well as for
  - The development of common criteria protection profiles.
- Followed closely by formation of working groups for:
  - Server, PDA, mobile phone platform specific implementation specifications.

TCG specifications

- TCG TPM main specification (general platform specification) version 1.2:
  - Design principles.
  - Structures of the TPM.
  - TPM commands.
    - (superseded TCG main specification version 1.1).
- TCG software stack specification version 1.1.
- TCG software stack specification header file.
- TCG PC specific implementation specification version 1.1.

Trusted Platforms: TCG definition

- A trusted platform:
  - A computing platform that has a trusted component;
  - Usually in the form of built-in hardware which is used to create a foundation of trust for software processes.
- Two fundamental functionalities:
  - ***Attestation:*** Third parties can measure PC integrity, e.g. using the *DAA protocol* (part of TCG specifications);
  - ***Secure storage:*** PC can be trusted to store information securely and only release it to certain applications.

Trusted Platform functionality (1)

- Trusted platform technologies aim to provide:
  - Confidentiality and integrity of application code and data;
  - Confidentiality and integrity of application code and data during storage;
  - Integrity of the operating system and underlying hardware such that the properties above can be satisfied.

Trusted Platform functionality (2)

- Platform authentication to external entities.
- Trusted path to user ensuring confidentiality of user input.
- Secure channels to devices and between applications to ensure confidentiality, integrity, and authenticity of inter-application communication.
- Ensure reliability by restricting size of trusted critical components:
  - Common estimate: 1 security-related bug per 1000 lines of code.

NGSCB

- *Next Generation Secure Computing Base* (NGSCB) is Microsoft's take on Trusted Computing.
- Version of Windows that uses trusted hardware (e.g. hardware conformant to TCG specifications) to build a trusted kernel.
- Allows trusted applications to run under control of a trusted operating system, in parallel to 'regular' Windows applications.

LaGrande

- Set of enhancements to Intel chip sets incorporating everything needed to build a Trusted Computing Platform.
- Also provides a potential platform for NGSCB-enabled PCs.

Using trusted computing

- It seems plausible that such technology – some proprietary, some standards conformant – will be included in most future computing devices (PDAs, notebooks, phones, …)
- Many applications for such technology have been proposed, most controversially for DRM.
- We consider how it might be used to address security issues in a ubiquitous computing environment.

What can we use it for?

- Possible applications relating to identification/authentication include:
  - stable identities for mobile devices;
  - single sign-on (SSO) implemented on a mobile/personal computing platform;
  - control of transfer and use of personal/contextual information;
  - verification of correct (unselfish) performance of a protocol requiring co-operation, e.g. for MANET routing.

TC based stable identities

- A major problem in scenarios lacking security infrastructure is the Sybil problem (entity claims multiple addresses) – e.g. in p2p and ad hoc settings.
- Trusted computing may be able to help by using the DAA protocol in a way which enables all actions of a particular platform to be linked, while not revealing true identity of that platform.

TC based trusted download I

- Problem arises in the context of broadcast to a mobile device.
- The established standard security techniques, e.g. involving use of broadcaster-specific smart cards in set-top boxes, is not really appropriate for a mobile model.

TC based trusted download II

- Use of a mobile platform for receiving broadcast content seems to mandate a software solution.
- However, the conditional access application needs to be protected.
- Conventional operating systems cannot provide the needed protection without a hardware token.
- Trusted computing can provide all the necessary guarantees.

TC based identity management I

- A range of different single sign-on (SSO) technologies exist.
- In a *true SSO* system, a user authenticates once to an Identity Service Provider (ISP), and this ISP then vouches for the identity of the user to multiple Service Providers (SPs).

TC based identity management II

- Clearly, the SP must trust the ISP to tell the truth about who has been authenticated and how.
- Typically this means that the ISP must be a networked entity remote to the user.
- The use of trusted computing technology enables the ISP to be implemented on the user platform, in such a way that the SP can verify its trustworthiness (using the *attestation* functionality).

TC based PI management I

- A growing number of possibilities now exist for Internet SPs to offer services tailored to end users.
- However, this possibility also represents a privacy threat, since the SP will typically need to know potentially privacy-breaching information about an end user in order to provide the tailored service. One key example (relevant in a mobile context) is the use of location information.
- One partial solution to the problem of controlling personal information (PI) is to attach policy info.
- However, such a system needs enforcement. Of course, part of that is regulation.

TC based PI management II

- It is possible that trusted computing can help.
- The holder of PI, and associated policy information (e.g. defining user preferences), can use trusted conmputing functionality to check out the platform requesting PI, before sending it.
- This check could involve verifying the type of recipient platform and the identity of the receiving application.

TC-based co-operation enforcement

- The support of MANETs typically requires co-operation by the nodes, e.g. to support routing.
- Malicious users may replace their network software with a 'selfish' version, e.g. to save battery power.
- TC could help guarantee that a network element is running the 'correct' software, and hence will not behave selfishly.
- (Of course, this requires the communications hardware to be part of the TC subsystem.)

TC-based type approval

- The spread of computers everywhere (cars, fridges, toasters, …) gives rise to major problems regarding safety.
- For example, a car owner could replace the engine management software to radically increase engine power. Similar problems arise with SDR.
- Not only will this potentially wreck the engine, it may also be a major safety problem, since the brakes/suspension won't match performance.
- Traditional solution is a closed environment which will only run authorised software – however the trend is to open platforms everywhere, and TC may help give back control.

Agenda

- Security – definitions
- Security and privacy
- Ad hoc relationship establishment
- Protocol and layering issues
- Key management
- Single sign-on
- Authentication and trusted computing
- Mobile code authorisation

Mobile code security

- Mobile code, i.e. executable code which is downloaded from one device to another, has obvious associated security issues.
- Main issues are:
  - Integrity of code itself;
  - Authentication of origin of code;
  - Authorisation of code.
- Other issues:
  - Code confidentiality;
  - Intellectual property issues.

Integrity and authentication

- These issues appear to be relatively easy to solve using existing technology.
- The ‘well known’ solution is:
  - Digitally sign the code before transfer;
  - Recipient verifies integrity of code using signature
  - Recipient obtains public key to verify signature by using a public key certificate.

Existing solutions

- This type of solution is already widely used.
- For example
  - Use of signed applets by web browsers.
  - MExE (Mobile Execution Environment) uses signed code.
  - Many proposals for mobile agent security.
  - Proposals for Software Defined Radio (SDR).

Issues

- Source and validity of ‘root certificates’.
- E.g., problems arise with web browsers where it is difficult to check and remove root certificates, and there is no simple method for revocation.
- Not clear whether this is a technology issue, or simply a question of waiting for existing ‘certificate status’ solutions (e.g. CRLs, OCSP) to be deployed.

Authorisation – the real problem

- Probably the trickiest issue is that of authorisation.
- On receipt of mobile code, and having checked its integrity and origin, do you execute it or not?
- Two decisions to be made:
  - Do I execute code from this source?
  - Do I execute this particular piece of code from this source?

Existing solutions

- Web browsers – on receipt of signed applets the authorisation decision is left to the user (who is encouraged to accept all code with a verifiable signature).
- MExE has a slightly more sophisticated approach, involving the provision of three different execution domains for code from different origins.

Where is this a problem?

- Everywhere processors are used!
- The problem of receiving and authorising mobile code arises in many domains, e.g.
  - Mobile phones,
  - Wireless-enabled PDAs,
  - Other mobile appliances.
- Software Defined Radio, where software can change how the radio interface works, creates even more problems, since regulatory issues are involved.

Why is this a problem?

- In the 'world of PCs' we already have a solution – it is up to the user to decide what software he trusts.
- End result: massive propagation of viruses, and unreliable devices (PC operating systems that need to be routinely re-installed).
- Do we want our mobile phones suffering similar problems (viruses already exist for some mobile devices)?
- What about other processor-based devices, e.g. cars?

It is not just about choice …

- Is this all just alarmist – why shouldn't users download code if they wish?
- Most PCs are configured to allow code to be downloaded automatically by web sites they visit. Typically this code is downloaded and executed with minimal user intervention (just a dialogue box).
- This is a huge vulnerability – most users don't wish to make such decisions.
- How do we help them, without hindering the 'expert user'?

General solution

- Ultimately we need a general automated solution to the authorisation problem, of following type:
  - Device contains a policy statement
  - Code contains a policy statement, and perhaps some authorisation permissions'.
  - Device authorisation control engine compares policy statement, assesses certificate chains, and makes a decision.

More reading I

- C. J. Mitchell (ed.), *Security for mobility*, IEE, 2004.
- Three recent Royal Holloway PhD theses:
  - E. Gallery, *Authorisation issues for mobile code in mobile systems*, 2006;
  - A. Pashalidis, *Interdomain user authentication and privacy*, 2005;
  - P.-W. Yau, *The security of routing protocols for ad hoc networks*, 2006.

  Available at http://www.ma.rhul.ac.uk/techreports

More reading II

- Zeroconf Working Group, Internet Engineering Task Force (IETF), available at:
  http://www.ietf.org/html.charters/zeroconf-charter.html
- T. Aura. *Cryptographically Generated Addresses (CGA)*. RFC 3972, IETF, March 2005.
- T. Aura. *Mobile IPv6 Security.* In *Proc. Security Protocols, 10th International Workshop*. LNCS **2845**, pp.215-228, Springer-Verlag, Berlin, 2003.
- J. Arkko, T. Aura, J. Kempf, V.-M. Mäntylä, P. Nikander, and M. Roe. 'Securing IPv6 neighbor discovery and router discovery'. In *Proc. '02 ACM Workshop on Wireless Security (WiSe)*, pp.77-86, ACM Press, 2002.
- P. Nikander, 'Denial-of-service, address ownership, and early authentication in the IPv6 world'. In B. Christianson et al., eds: *Proc. Security Protocols, 9th International Workshop*. LNCS **2467**, pp.12-21, Springer-Verlag, Berlin, 2001.

Acknowledgements

- Must thank all my research students (past and present), who have worked on many of the problems discussed, and who have contributed to this presentation.
- Particular thanks to:
  - Eimear Gallery;
  - Adrian Leung;
  - Andreas Pashalidis;
  - Po-Wah Yau.

Questions

- For further details on any topics addressed please contact me:
  - c.mitchell@rhul.ac.uk
  - http://www.isg.rhul.ac.uk/~cjm
  - Chris Mitchell
Information Security Group
Royal Holloway
University of London
Egham, Surrey TW20 0EX
UK