Security Without Authentication?

Geraint Price

Information Security Group

Royal Holloway, University of London

Presentation Outline

- Introduction
- Client Puzzles
- A Real World Example: CAPTCHAs
- Authentication: Where? Why? How?
- Possible Solutions
- Conclusions

Introduction

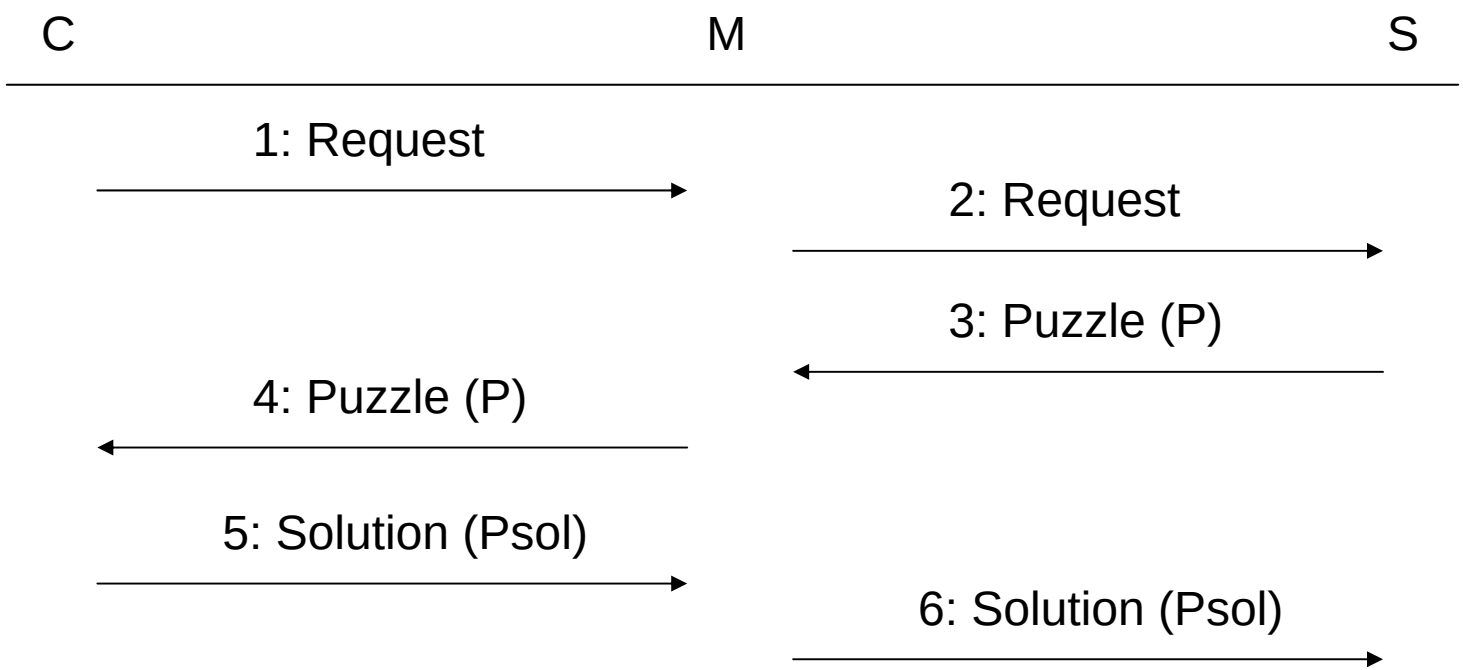
- Interest in this area after doing some research on Denial of Service attacks.
- What assumptions are made when people are designing security for distributed systems?
- What applications and solutions require authentication?
- Can we, as security engineers, justify the use of authentication as an underlying primitive?
- What happens if we cannot?

Client Puzzles – I

- Client puzzles were proposed as one means of preventing a Denial of Service (DoS) attack.
- Basic principle:
  - A client requests a resource.
  - The server returns a puzzle which the client is asked to solve.
  - The server is only willing to commit the resource once the client has spent some effort solving the puzzle.
- A legitimate client is willing to accept a slight delay, but the accumulative delay makes the cost of an attack prohibitive.

Client Puzzles – II

A generic weakness:



The weakness comes from the lack of binding between an instance of a Puzzle and the legitimate participants of the protocol.

CAPTCHAs

- **CAPTCHA**: **C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part.
- Usually a picture, or set of pictures, with a query that humans can easily answer, but which are difficult for a computer to answer.
- In Jan'04, it became apparent that spammers were using a similar attack to ours to bypass the captchas.
- Spammers set up websites that handed out free porn in return for clients solving captchas forwarded from free email sites.
- Again, a lack of binding authentication.

Some Qualifying Statements…

- We are generally only concerned with widely distributed environments.
  - It should be easier to construct authentication services in a centralised system.
  - How do you find even the simplest of anchor points in a widely distributed environment?
- Transitory interactions amplify this problem.
  - Examples where the wide-scale nature of the environment is no hurdle: EMV.
- We face the biggest problems when we migrate services from a *local* to a *remote* solution.

Who Needs Authentication?

- A traditional view of implementing security is in the provision of *services*.
- What other services rely on an authentication service:
  - Confidentiality?
  - Integrity?
  - Availability?
  - Auditing?
  - …

What Are We Protecting?

- Gollmann notes the fact that the original Java *sandbox* model did not rely on authentication.
- Can we protect the information by protecting the means by which we store/process the information?
- If we consider a transaction processing environment:
  - Can you undo a transaction easily?
  - What's the penalty for committing a non-recoverable transaction?

Possible Solutions

- What is needed is a “toolkit” comprising of different solutions to this problem.
- Depending on the application requirements, we should either:
  - decide that authentication is required, and pay the price;
  - decide that authentication is not required and do away with it;
  - design the resulting solution accordingly.

A Risk Management/Restricted Transaction Model

- For example, a banking application:
  - Soft certificates were used at the client end.
  - The level of trust in the key was greatly reduced.
  - Thus, there was a restriction on the types of transactions which the remote client could execute.
- A potential problem with this is that it relies on a fairly intuitive grasp of *risk*.
- Traditional risk management does not translate well into information security.

A Fault Tolerant Approach

- Adapting fault tolerant mechanisms for security has been part of the security literature for a while:
  - Rampart
  - MAFTIA
- To date, most of the work makes use of voting mechanisms – but this relies on Authentication…
- What other mechanisms can use?
  - Timing measurements are one example which we have previously considered.
  - Can we further adapt mechanisms developed for other types of intrusion prevention (e.g. smart card techniques)?

Trusted Computing

- Allows remote entities to check the state of a trusted component on your machine.
- Provides protected hardware for key storage.
  - Solves one of the problems when using cryptographic mechanisms.
- Uncertain how the mapping from hardware modules to individuals will happen.
  - This will impact on the types of services which can then rely on the authentication.

Ad Hoc Networks

- If you do not have an infrastructure to rely on, then you are forced to be more inventive.
- Stajano's and Anderson's work: *The Resurrecting Duckling*.
- Should the Ubicomp community rely on the mechanisms designed for long lived/static structures?
- Most of the designs for those structures do not even work in their intended domain!
- Design solutions that do not rely on an infrastructure.

Self-Certification

- In the security community self-certification is usually frowned upon.
- However, it might be of use when all you want to do is bind an action to an entity…
- … without caring *who* that entity is – as in the case of client puzzles.
- This could work in situations where you do not require an audit or non-repudiation service.

Electronic "Footprints"

- Credit reference agencies often use electronic biographical *footprints* to authenticate individuals.
- Can such footprints be developed in the virtual world?
  - Ebay uses ratings for their users.
- How would you measure them?
- How you track them?
- What happens if they are compromised?
  - *Virtual Identity Theft*.

Some Potential Design Principles…

- Do not rely on any underlying infrastructure.
  - Just because it is tangible, it does not make it trustworthy.
- Understand what it is you are protecting and why.
  - Does it help if you are protecting the mechanisms rather than the information?
- If all else fails: use a cryptographic key.
  - **WARNING**: If you do not know how to get the keys to the principals, then your design is likely to fail.

Related Issues

- Microsoft's identity initiatives:
  - Passport to InfoCard.
- De-Perimeterisation:
  - With the breakdown of the security perimeter, the difficulties faced are likely to become more pervasive.
- Intrusion Detection/Prevention:
  - IDS/IPS and firewalls have to make do with imperfect rule sets.
- Provable security:
  - Proving properties of protocols is hard without the notion of *who* is running the protocol.

Conclusions

- Clearly, we cannot ignore authentication altogether.
- We advocate a more judicious use of authentication (and hence the reliance on cryptography).
- What are we trying to protect?
- Most security measures are not robust – we, as a research community, should rectify this.

Q & A…

Thank you.

Geraint Price
geraint.price@rhul.ac.uk