Distributed Proctoring

Bruce Schneier * John Kelsey ** Jay Walker ***

Abstract. We develop a protocol for "distributed proctoring" which allows a network of graders to grade individual problems solved by a network of test takers. The mutual anonymity of the test takers and graders is ensured using anonymous MIXs, but an audit trail is provided in the event of a grading dispute. We discuss other applications for this protocol, more generally defined as "digital piecework."

1 Introduction

We would like to take some submission from a user (e.g. a contest entry, a solution to a problem in an exam, or a paper submitted to a refereed electronic journal) and pass it anonymously to be judged by one or more graders. We want the graders to be able to send an authenticated result back, and we need to be able to pay them for their work. Eventually, we will make some decision on the basis of this result: an exam is given a grade, a paper accepted or rejected, a winner declared in a contest. We call this general problem "Distributed Proctoring."

There are many applications for this kind of protocol, and it can be more generally thought of as "Digital Piecework." Alice has a problem requiring human judgment (grading test answers, in the example above) which can possibly be divided into several pieces (each test question can be graded individually). There is a network of experts (graders) who are available to solve problems (they log onto the system at will, and are sent test answers to grade as they become available). The experts solve the problems (grade the questions) and transmit their solutions back to Alice. This system could be anonymous or not. Alice needs to be able to verify the experts' credentials (they must post certified copies of their credentials) and possibly their identity, and some means of paying them for their work.

There could be an entire network devoted to digital piecework. Experts in a variety of fields could post their credentials, and people needing expertise could post their requests. An "experts wanted" list might look like:

– I want an academic paper on quantum mechanics reviewed anonymously by three qualified academics not located in the U.S. The reviews are due on a particular date.

* Counterpane Systems, 101 East Minnehaha Parkway, Minneapolis, MN 55419 schneier@counterpane.com

** Counterpane Systems, 101 East Minnehaha Parkway, Minneapolis, MN 55419 kelsey@counterpane.com

*** Walker Digital, Inc., 4 High Ridge Park, Stamford, CT 06905

- I am a businessman in the steel salvage business with a question as to how to handle a personnel problem. I would like my question answered by a personnel professional at a Fortune 1000 company who is not in the steel business and who has dealt with the same or similar problem at least five times in their own company. I am willing to pay $100 for answers or at least 250 words. Anonymity is acceptable, but I will pay an extra $100 if the person is identified to me and I am allowed to call him.
- I am an attorney who is seeking other businesses who have had a problem with the XYZ company's plastic products. I will pay $50 for each documented example of problems, up to 100 problems.
- I am looking to decode a hieroglyphic unearthed on a site in Israel. I will pay $50 for each solution of at least ten words, and $500 for the complete solution. Answers will be accepted from 4 candidates who send me their credentials.
- I am seeking to hire a qualified structural engineer for a two-week assignment performing a weight-load analysis for various wooden bridges.
- I am arguing with an insurance company, and need three independent medical opinions no later than the following Friday. I seek only the opinions of doctors who are board-certified inner ear specialists and who have been practicing for five or more years. Each opinion must be at least 200 words in length. If the opinion is anonymous or unofficial, I'll pay $100. If the opinion is the physician's official opinion, I'll pay $250.
- I am looking to find Alice Smith, last known address 100 York Street, Anytown, USA. I will pay $500 to the first person who provides me with a verified place of employment for this person.
- To all people who have eaten at my restaurant, I will pay $5 for a 100 word description of how to improve my service or menu: limit, 50 people.

For the rest of this paper, we will restrict our discussion to the distributed proctoring problem. Many security issues arise in this kind of protocol. Among them:

a. How do we know that the test's rules are being followed?
b. How do we know that the grader is grading tests impartially and correctly?
c. How do we ensure anonymity of the test takers and graders?
d. How do we prevent collusion between some test takers and some graders?
e. How do we prevent outside interference with the process?

1.1 Notation and Conventions

a. $PKE_{PK_U}(X)$ represents the public-key encryption of X under public key PK_U.
b. $SIGN_{SK_U}(X)$ represents the digital signature of X under private key SK_U.
c. $E_{K_0}(X)$ represents the symmetric encryption of X under key K_0.
d. PK_U represents the public key of user U.

e. SK_U represents the private key of user U.
f. ID_U represents the identification number or code of user U.
g. X, Y represents the concatenation of X with Y.

2 Techniques

2.1 Anonymity and MIXs

An "anonymous MIX" is an intermediary communications node which attempts to make it very difficult for anyone to trace the path of a message which passes through the MIX. The idea behind MIXs is discussed in [Cha81], and is used in election schemes [Cha81, PIK94] and anonymous remailers on the Internet [Sch95, TG96]. In our protocol, we use anonymous MIXs to prevent an eavesdropper who has access complete access to the network's traffic from being able to compromise the system.

When Alice sends Bob a message through an anonymous MIX, the following (roughly) takes place:

1. Alice wishes to send message P to Bob anonymously. She first forms
   K_0 = a random session key.
   Z_0 = an all-zero string of some random length.
   $X_0 = PKE_{PK_M}(K_0)$, where PK_M is the MIX's public key.
   $M_0 = X_0, E_{K_0}(ID_B, Z_0, P)$, where ID_B represents Bob's identity.
   She then sends M_0 to the MIX. Note that Alice may also have encrypted and digitally signed the message she's sending to Bob. This has no bearing at all on how the MIX processes it.
2. The MIX receives M_0. It decrypts this into ID_B, P. It looks up Bob's public key from ID_B, and then forms
   K_1 = a random session key.
   Z_1 = an all-zero string of some random length.
   $X_1 = PKE_{PK_B}(K_1)$, where PK_B is Bob's public key.
   $M_1 = X_1, E_{K_1}(Z_1, P)$.
   It waits some random amount of time before sending M_1 to Bob. During this time, it is processing many other messages, both sending and receiving them.
3. Bob receives M_1. He decrypts it and recovers P.

In order to anonymize messages that pass through a MIX, we must have a fairly large volume of messages coming in and out and a random delay involved in forwarding those messages. Otherwise, it is possible for an opponent to watch and time messages going into and coming out of the MIX, and use this information to determine the source and destination of each message. The whole point of the strings of random-length all-zero string (it can be a random-length string of any random bits, but there needs to be some way to determine where Z_i ends and P

begins) is to disguise the size of the messages; an eavesdropper who watches the messages coming in and out of the MIX cannot link them by their size. Messages may also be broken into many pieces to avoid this problem. Similarly, messages must be encrypted to the MIX, so that the messages can be decrypted and then re-encrypted with a different key.

2.2 Steganography, Marking, and Collusion

There exist techniques for embedding a hidden message in almost any free-format data. This can be used to allow Alice to mark her submission in some invisible way, so that if Bob is colluding with her, he can identify her submission. Given that submissions generally need to be free-format, there is not a general cryptographic solution for this problem.

The study of how to embed invisible marks and messages in other messages is called steganography. Many such techniques of varying quality are well-publicized. However, one general rule is that it is very hard to eliminate any chance of such hidden information without radically altering the "covertext" data. However, proper design of the tests may help make it hard to insert covert signals into the submission in practice. Mainly, however, we rely on the difficulty of making contact with the graders and bribing or threatening very many of them into co-operating, to make this attack less threatening.

3 The Protocol

In this protocol, the players are:

- Alice — the user submitting something to be judged.
- Bob — the person judging Alice's submission.
- Carol — the person coordinating the whole thing. She is completely trusted by all parties.
- Dave — a person who may want to have Alice prove the results to him.

We use public-key cryptography for encryption and digital signatures. The exact algorithms are unimportant at the protocol level; an overview of the techniques involved is [Sch96]. All public keys are signed by some certification authority. Certificates can be sent with messages, and different keys can be used for encryption and digitial signatures. Carol knows everyone's public key, and everyone knows hers. The anonymous MIXs either know everyone's public keys, or their public keys are sent along with their identities. Everyone is assumed to know the MIXs' public keys.

3.1 The Basic Protocol

1. Alice creates some submission, S. This may be in response to a question from an exam booklet, or a contest, or a post submitted to a moderated newsgroup, or whatever. She then forms

K_0 = a random session key. This should not be confused with the key Alice uses to send the message to the anonymous MIX.
R_0 = a random challenge either generated by Alice or given to her by someone else, depending on application. In some cases, R_0 is used to seed the test machine.
$X_0 = PKE_{PK_C}(K_0)$, where PK_C is Carol's public key.
$X_1 = SIGN_{SK_A}(ID_A, R_0, S)$, where SK_A is Alice's private key and ID_A represents Alice's identity.
$M_0 = X_0, E_{K_0}(ID_A, R_0, S, X_1)$.

She sends M_0 to Carol via an anonymous MIX.

2. Carol receives M_0. She decrypts it, and verifies the signature. She also verifies that she's never seen this submission before. She randomly selects a Bob out of all the graders available. She then forms

K_1 = a random session key.
N = a random submission identifier.
T_1 = a timestamp.
$X_2 = PKE_{PK_B}(K_1)$, where PK_B is Bob's public key.
$X_3 = SIGN_{SK_C}(ID_C, N, T_1, I, S)$, where SK_C is Carol's private key and ID_C represents Carol's identity, and where I are the grading instructions.
$X_4 = E_{K_1}(ID_C, N, T_1, I, S, X_3)$
$M_1 = X_2, X_4$.

She sends M_1 to Bob via anonymous MIX. She stores N, ID_A, R_0, S, and T_1.

3. Bob receives M_1. He decrypts it, and verifies the signature and timestamp. He then grades the submission, S, according the the instructions, I. When this is finished, he has formed a grade, G. He then forms

K_2 = a random session key.
$X_5 = PKE_{PK_C}(K_2)$.
$X_6 = SIGN_{SK_B}(ID_B, N, G)$, where SK_B is Bob's private key and ID_C represents Bob's identity.
$X_7 = E_{K_2}(ID_B, N, G, X_6)$
$M_2 = X_5, X_7$.

He sends M_2 to Carol via anonymous MIX.

4. Carol receives M_2. She decrypts it and verifies the signature. She searches for the matching N among her currently active submissions. If she doesn't find it, an error has occurred, and someone will need to follow up, but the protocol ends. If she does find it, however, she notes that Bob has returned the submission (so he can be paid for it). She then looks up Alice's challenge, address, and timestamp, and forms

K_3 = a random session key.
T_2 = a new timestamp.
$X_8 = PKE_{PK_A}(K_3)$, where PK_A is Alice's public key.
$X_9 = SIGN_{SK_C}(ID_A, R_0, T_2, S, G)$.
$M_3 = X_8, E_{K_3}(ID_A, R_0, T_2, S, G, X_9)$.

Carol sends M_3 to Alice via anonymous MIX.

5. Alice receives M_3, and verifies the signature. She now has an authenticated grade on this submission, along with a timestamp to show when she submitted it.

In step (1), Alice encrypts the submission under the public key of Carol, the co-ordinator of the proctoring system. This gives Alice some confidentiality (important for some kinds of tests), makes it harder for others to copy Alice's submission, and also prevents an eavesdropper from seeing Alice's specific submission, and thus being able to easily trace it.

Alice also signs the submission under her own private key. This prevents someone from changing Alice's submission, either to help or hurt her score, without knowledge of her private key. If we are concerned that Alice may give or lose her private key to someone else interested in changing her submission, then we can also have the test-taking machine digitally sign the submission with a key that's kept in tamper-resistant storage somewhere.

Alice incorporates a random challenge into her submission. This allows another party, Dave, to verify that she's actually getting these results, without having to look over her shoulder and verify her score.

In step (2), Carol generates and uses N, an anonymous identification code for this submission. This allows her to efficiently cross-reference between submissions already sent out to graders, and those waiting for a return from the graders. Carol needs to verify that she has never seen the submission before, in order to prevent Alice from submitting multiple S values and keeping the one with the best grade. In some applications she can check R_0; in other applications she needs to keep a database of which exam questions Alice has already answered.

Carol's timestamp prevents simple replay attacks. Her public-key and symmetric encryption prevents loss of confidentiality of submissions, and also to complicate attempts to figure out which submission went to which grader. Finally, her digital signature ensures that graders don't grade bogus submissions, and that nobody can change any of these submissions without being noticed.

In step (4), Carol signs the message; this authenticates the grade for Alice and Dave, as well as ensuring that no malicious or accidental changes to the grade have occurred. Carol encrypts the message, ensuring Alice's confidentiality. And Carol includes both a timestamp (from when the submission was received) and R_0. This allows Alice and Dave to be certain that this isn't simply a replay of someone else's successful test result. It also contains S, so that Alice can immediately notice any changes that have occurred in S.

At this point Alice must also verify that she expected X_7 from Bob. Otherwise, if N were easy to guess, Alice could send the grade that she wanted to Carol before Bob did. Or another grader could send a grade to Carol for a pending submission and receive pay, even if that grader was not assigned to the submission.

3.2 Enhancements and Variations

Showing Results to Someone Else If Alice is making a submission whose result she must show Dave, then she gets R_0 from Dave. In this way, she is able to prove that there is no replay going on.

Preventing Signaling by Dave If Alice is concerned that Dave might pass use R_0 as a subliminal channel [Sim85, Sim94] to pass information to Carol, they can mutually agree on R_0. The simplest way is probably as follows:

1. Alice generates random R_1, and sends Dave $hash(R_1)$.
2. Dave sends Alice R_2.
3. Alice generates $R_0 = R_1 \oplus R_2$.
4. Alice makes her submission, and receives her answer back.
5. Alice sends Dave R_1 and the response she received from Carol.

Auditing the Performance of the Graders Sometimes, it may be important to judge the graders' competence and impartiality. In this case, the same questions or submissions can be sent to both a trusted grader and an untrusted one. Then, the answers can be compared. Alternatively, each submission might have some chance of being sent out to more than one grader. Significant disagreements would cause Carol to review both grades, and possibly to determine that one grader was being partial or incompetent.

3.3 What Can Go Wrong?

- The anonymous MIXs may fail, making it possible for a network eavesdropper to determine which graders graded which submissions. This opens up possibilities for all kinds of potential problems, especially involving bribery or blackmail of the graders. If Carol adds random delays in steps (2) and (4), she can act as an anonymous MIX herself, though it would still be preferable to use a larger set of anonymous MIXs as well. Additionally, she may wish to agree ahead of time with some of the graders and test-takers, or with others, to send and receive "null messages," which look like real messages until decrypted. This could happen either because the MIX machines are subverted, or because they don't have enough traffic, don't have random enough delays, etc.
- If the public key or symmetric encryption algorithms are broken or badly implemented, then the anonymity of the graders and users can be compromised.
- If the digital signature or hashing algorithms are broken or badly implemented, then it will be possible to carry out active attacks in which Alice's submission is replaced with a better (or worse) submission.

– Some test-takers may conspire with some graders to get unfairly high grades. In most cases, this can be done using some steganographic techniques to embed an identification string into the submission, and possibly even into the grade. The only general solution to this seems to be to make sure that the graders are audited occasionally, and to ensure that most graders are honest.
– Graders may be incompetent or dishonest, and not do a thorough job grading.
– Depending on the way graders are rewarded for their work, it may be possible to acquire a list of all the graders. This would make blackmail or bribery far simpler. Where possible, anonymous payment methods should be used.

4 Applications

There are many possible applications based on these ideas. Three are discussed briefly in this section.

4.1 Distributed Proctoring of Exams

Distributed proctoring of exams simply allows a user to take an exam under some simple scrutiny to prevent cheating, and then allows that exam to be graded by people who have never met the user and probably never will.

The protocol discussed in subsection 3.1 is directly used for this application. Alice's instructor, Dave, gives her random challenge R_0. She responds, after taking the exam, with the response returned to her by Carol. Carol, of course, keeps a database to ensure that Alice only responds with one answer to each exam question.

One potential problem is that the graders may not be very consistent. Many people who grade exams or programs try to grade them all in one sitting to avoid treating one group of students differently than another. It may make sense to require the same grader to grade many exams, or the same problems from many submissions. Alternatively, it may make sense to have several graders judge each exam, and average out their scores, or even normalize test scores based on averages per grader.

4.2 Judging a Contest

Contests go on all the time. Some ask participants to sketch a suggested new company logo, or come up with a new slogan. Others might ask the participants to estimate some number (like the number of jelly beans that fit in a large room), or solve some puzzle (such as a very large and complex crossword puzzle.) Distributed proctoring in this case is used to impartially judge contest submissions.

Contest submissions are different from exams in that they are probably graded by several people in sequence, as the graders weed out all but the best submissions. The only major change in this application's use of the protocol in section 3.1 is that steps (2) through (4) may take place many times for the same submission. Each time, Alice is notified of how her submission has fared. Eventually, if Alice is the winner, she is sent her final notification. For judging a contest, Carol may send each submission to several judges at once, and then take the average of their grades as the true score. There may then be a final judgement between the most promising three or four submissions.

Along with the things discussed in section 3.3, there are other potential problems here:

- These contests may have monetary prizes. This raises a whole set of possible problems regarding payment protocols and such. Also, many attacks become worth considering when there is a large potential payoff for succeeding.
- Contests with a single corrct or best answer, or a straightforward judging criteria, would never need a set of human graders. However, the more subjective the grades are, the harder it is to determine whether a given grader is giving some submissions better treatment than others. This implies that judgements should probably be done using many different graders, and averaging their scores somehow.

4.3 Refereeing a Journal or Newsgroup Submission

Moderated Usenet newsgroups and internet mailing lists have existed for a while now, and refereed technical journals for much longer. The technical journal of the future will probably look like a marriage of the two. A distributed proctoring system is useful in getting a submission directly to one or more referees. Depending on the application, there may be several levels of refereeing between initial submission and final acceptance.

For most moderated newsgroups, the intent of moderation is simply to filter out irrelevant or inappropriate submissions. The moderators seldom (if ever) filter the submissions for quality and originality of technical work. The problem in moderating a newsgroup is dealing with volume.

Distributed proctoring works here because a large number of graders can each take part in moderating the newsgroup. Most newsgroup articles can be filtered for relevance and appropriate topic pretty quickly. If there is some method by which people pay to receive or use the newsgroup, then each moderator can be paid according to how many articles he grades. (The quality of his judgement will also come into play, probably in determining whether or not he will continue to be a moderator for the group.) The protocol described in section 3.1 is followed, except that in step (d), Carol posts Alice's submission to the newsgroup. In some cases, she also includes the grade. Note that the grade may be more about the post's topic (keywords, for example) than about its quality.

Technical journals are quite different. Most have a much more narrow audience, and they generally filter for technical quality of the submissions. Grading will take longer for a technical journal, and there will probably be several graders involved. Also, for a technical journal, it is quite reasonable for the graders to reject the submission for now, but make some suggestions for ways to make the submission acceptable at a later time.

Distributed proctoring works well here because the grading process needs to be anonymous, and because while there needs to be a thorough review of submissions, turn-around time is also important. For this application, the protocol in section 3.1 is modified in step (2): the submission is probably sent out to several graders. The final grade is based on all the grades collected: possibly a majority vote of accept or reject. However, all comments are passed back to Alice in step (4). It would not be unusual for Alice to alter the submission based on those recommendations before resubmitting it for final approval. Final approval means that in step (4) of the protocol, Carol forwards the submission to the journal's printing or remailing system, as well as to Alice.

Along with the things discussed in section 3.3, there are other potential problems here:

- Moderators or referees may not do a thorough enough job grading the submissions. For example, a dishonest newsgroup moderator might only read the first paragraph of a submission before deciding to accept or reject it. The auditing techniques discussed above should minimize this problem. Also, there will probably be a record kept of each grader's acceptances and rejections. There may even be a way to allow the subscribers to rate the submissions' quality, and thus effectively grade the graders.

5 Conclusions

There exists a need to efficiently match people with expertise with people requiring that expertize; exam proctoring is only one example of this. We have shown how to use anonymous MIXs to support this commerce in expertise. Potential applications include buying expert opinions, hiring of temporary, part-time, and permanent employees, and marketing research. The more diverse the applications that use the MIX protocol, the more security inherent in the system.

6 Acknowledgments

The authors would like to thank Steve Bellovin, Chris Hall, James Jorasch, and David Wagner for their helpful comments and suggestions. The digital piecework protocols are patent pending in the United States and other countries.

References

[Cha81] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM,* v. 24, n. 2, Feb 1981, pp. 84-88.

[PIK94] C. Park, K. Itoh, and K. Kurosawa, "Efficient Anonymous Channel and All/Nothing Election Scheme," *Advances in Cryptology — EUROCRYPT '93 Proceedings*, Springer-Verlag, 1994, pp. 248-259.

[Sch95] B. Schneier, *E-Mail Security*, John Wiley & Sons, 1995.

[Sch96] B. Schneier, *Applied Cryptography, 2nd Edition,* John Wiley & Sons, 1996.

[Sim85] G. Simmons, "The Subliminal Channel and Digital Signatures," *Advances in Cryptography — EUROCRYPT '84 Proceedings*, Springer-Verlag, 1985, pp. 51-57.

[Sim94] G. Simmons, "Subliminal Channels: Past and Present," *European Transactions on Telecommunications*, v. 5, n. 4, 1994, pp. 459-473.

[TG96] G. Tsudik and C. Gulcu, "Mixing E-mail with BABEL," *ISOC Symposium on Network and Distributed System Security '96*, to apprear.

This article was processed using the LaTeX macro package with LLNCS style