Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2830855 A1 2015/04/25

(21) 2 830 855

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2013/10/25

(41) Mise à la disp. pub./Open to Public Insp.: 2015/04/25

(51) Cl.Int./Int.Cl. *G06Q 20/08* (2012.01)

(71) Demandeur/Applicant:
CHOW, STANLEY, CA

(72) Inventeur/Inventor:
CHOW, STANLEY, CA

(74) Agent: NA

(54) Titre : CONCEPTION DE SYSTEME DE MICROPAIEMENT POUR CONTENU WEB
(54) Title: A DESIGN FOR MICRO-PAYMENT SYSTEM FOR WEB CONTENTS

Detailed design for Micro-payment system for web contents

By Stanley Chow, May 27, 2013 All rights reserved

Updated October 9, 2013 All rights reserved

Assumptions

- We will use X.509 certificates throughout, with all private keys being 2048 bits (this is projected to be safe till 2030 see http://en.wikipedia.org/wiki/Key_size)
- A reasonably fast hash like SHA-3 is used.
- With the recent NSA revelations, we may need to use longer keys, but the estimates don't change much.

Remaining work items (not part of patent)

- How different entities get and set up keys and other cryptographic stuff. These are not particularly difficult – we merely need to follow good practices. We expect key handling to follow X.509 models with the proviso that sub-PA will be signing many transaction and so the private-key must be available on the signing machine and that requires careful security design.
- Much of the security depends on the transactions being very small. A through analysis must be done before allow large value transaction.

The entities

- User – typically represented by a Web Browser
- Payer – this entity "injects" money into our system in that all payments originate from a Payer. A Payer is possibly the user's ISP, possibly an association that is packaging access to certain web sites. Represented by some Payer Portal that processes requests in real-time, this Portal will be signing of transaction any must be secured. Note that a Payer is also, sometimes, the receiver, in that a a Vendor will collect money through its Payer. Think of the Payer as the local bank branch.
- Payment Authority – this entity guarantees that the Payer will indeed pay. There can be multiple layers of Payment Authorities. Represented by some PA Portal that handles queries in real-time. This Portal will be mostly handling validation requests. As for Payer, each Payment Authority will handle both side of each payment. Think of Payment Authority as the National HQ of the local bank branch.
- Clearing House – this entity routes payments between Payment Authorities much like a cheque or credit card clearing house. There can be (in fact, usually will be) multiple Clearing Houses in any transaction. Represented by some Clearing House Portal in real-time.
- Path Finder – this entity is like DNS (Domain Name System). DNS takes a domain name (like micropebble.com and looks up the corresponding address 123.45.67.89). Path Finder takes two lists of Payment Authorities and find all the paths that can get from the first list to the second list.

- Vendor – typically a web site that is charging money for content.

Which Path-finder

At first blush, it would appear the Path-finder is a necessary evil to allow payments to happen, so none of the entities will want to provide this service. It turns out the Path-finder is the key to advertising, so whoever controls it controls making money. Our approach is to say everyone gets to provide this. The simplest approach is to let each entity provide a PFP (Path-finder Portal) and User will ask each Payer to get paths and parents, then ask parents to get more paths recursively. This is entirely workable but forces the users to do a lot of queries over the network to a lot of different entities; which mean it is likely to be slow.

The recommended approach is to have each Payer PFP return paths for itself plus its parent-PA and/or CA. (Since the users only see and deal with Payer's, this is the most convenient way for parent-PA and CA to insert their advertising).

Another recommended approach is to leaving adverting information out of the path, but each entity will provide a portal that the User can query for advertising deals.

Clearly, each Payer and its parent must have a protocol for updating the path information. We expect that, typically, the parent will do a daily download of path information (possibly including date ranges – this is especially useful for time-limited advertising).

Typically, each entry in a wallet will include the Payer responsible, the PFP information.

How many Paths

The path calculation may appear daunting, it is actually easy. Let's put some estimates (these are really loose estimates to give the idea of scale, some may be out by factors of ten or hundreds) of the different entities:

- National Clearing Houses – these are more or less "national" in scope, but with competition so there may be a few per country. Say there is a thousand.
- Specialist Clearing Houses – these are the loyalty points and the like, say another thousand.
- PA – these are equivalent of the big banks, or the big consortiums, probably in the neighborhood of hundreds in majors countries and a few in small countries. Totaling ten thousand world wide.
- Payer – these are equivalent of the ISP or local bank. Assuming we have a billion uses, we are unlikely to have more than a million sub-PA (otherwise, may sub-PA will be handling very few users, making them uneconomical).

Since paths are from Payer to Payer, assuming a million Payer's, we are looking at a trillion endpoint pairs, plus the multiplicity of paths (due to currency, etc) between each pair of endpoints. Potentially, we are looking at hundreds of trillions of paths.

Fortunately, each Payer has a much simpler problem – it only deals with paths with it-self as the starting point! It also only needs to handle one currency!! This means each Payer will only need to worry about the paths to get to each far-end Payer.

If the Payer stores these paths on a relational database, say a table indexed by the far-end Payer, with each row being the list of paths to that Payer. Each paths is likely to be under ten entities, each entity takes say a hundred bytes to describe charging, etc. Making each path be under 1K bytes. If we assume, very generously, there are ten paths to each far-end, we are looking at 10K bytes per row. So the table will have a million rows, each of 10K bytes. The whole thing is around 10GigBytes.

By modern standards, this is a small database (but probably still requires tuning for optimal performance).

Calculating Paths

Each Payer (or any other entity) can pre-calculate paths to all end-points and pre-populated the table (say every night after it receives the daily download from its parent PA). It is also possible to calculate the paths on-the-fly and cache the result for subsequent queries. Each Payer can make the choice independently.

We also expect that in some cases, the data transfer can be substantial – e.g., asking a PA for its list of Payers could be multi-megabytes. We expect the protocols to have an "update from yesterday" or "update from this-date-time" feature (this is an optional optimization).

We solve the problem hierarchically:

- The Clearing Houses are expected to know about each other and the connectivity between each other. This is reasonable since this information does not change often and there are few CH.
- Each Payment Authority is expect to calculate paths to all other PA. This is done by:
  - ask parent CH for CH connectivity/paths
  - construct reachable set of CH
  - for each reachable CH, ask for list of PA
  - for each PA in returned list, append PA to path and store
- Each Payer is expected to calculate paths to all other Payer. Note that this step is unnecessary if we include PA information in coupons). This is done by:
  - ask parent PA for paths to all other PA.
  - For each reachable PA, ask for list of Payer
  - for each Payer in returned list, append Payer to path and store.

The message sequence

1. User requests information from Vendor
   1. something like "get item-URL" via HTTP
   2. "item-URL" identifies the exact item of purchase
   3. recall that HTTP specifies return IP, desired language[s], desired format[s] and other information
2. Vendor sends back price quote
   - In normal web usage, a "HTTP get" request will get a status code of 200 – meaning OK. Sometimes, a status code of 401 – meaning unauthorized (along with the WWW-Authenticate Header field) and the user should retry the request with the necessary authentication.
   - HTTP in fact has a status code of 402 – reserved for Payment Required. We can either use 402 (which requires updating the standard) or use 401 (and define our own

convention of how the WWW-Authenticate field is structured).

- In either case, the price quote will include:
  - a list of prices in different currencies. Note that the prices for each currency may be wildly different – indeed, some currencies (like airline frequent flyer points) may not convertible to other currencies. Some prices can be strange – "free, but require membership" or anything else.
  - A list of Payers known to the vendor, including with currencies can be handled by which Payer (and its PA).

3. User asks Path Finder(s) for paths. This purpose of this step is to find a chain of PA and/or Clearing Houses that starts with the User and ends with the Vendor.
   - The User receives the payment request (more strictly, the Browser program receives the request on behave of the user)
   - User will have a "wallet" with a list of Payer (may be the user's ISP, a professional association, a specialized aggregator like model trains), also including the currencies handled by each Payer.
   - Send both lists of Payer+currencies to the Path Finder (along with User ID and whatever information the User-Payer will want) [This assume the Path-finder function is performed by Payer; if other entities are allow, adject accordingly]
4. Path Finder sends back viable paths.
   - look up (or construct) all possible paths between each User Payer and each Vendor Payer
   - each possible path must include necessary currency conversion
   - each component in each path must include the "handling fee" charged by that component. Ideally, the handling fee should be expressed as "+0.2 cents" and/or "+4%" but there is nothing to prevent a more complex language, we are likely to start with just the above two forms plus an escape to a URL. The Escape URL is intended to do things like signing up for a news letter and can return a return code indicating success or not. The Escape URL should also have a text description of it does, for example "free if you sign up for news letter" would be displayed as the cost of this path. When user picks this path, the Escape URL is invoked to sign the user up for news letter. After the Escape URL returns a magic token, the token is passed to the vendor as proof of signing up.
   - each possible path should be (cryptographically) signed – we only care about integrity and authenticate, so we only need it signed and no need to encrypt.
   - Preferably, the paths can be pre-computed and pre-signed. (***** for simplicity and speed, we may want to break a path into several signed pieces – Payer-to-CH, CH-to-CH, CH-to-Payer. ******). This minimizes CPU time for Path-finder, but means actual cost calculation must be done by user.
   - possibly insert additional paths for third party advertising and such
   - send list of possible paths back to User
5. User calculates cost of each path
   - After the User had the list of possible paths (either from Path Finder or Clearing Houses), go through each path to evaluate "actual cost" and possible advertising deals.
   - A path is something like: User → Payer-Verizon → PA-CitiBank → CH-USA → CH-Canada -> PA-RoyalBank → Payer-RB-Vancouver → Vendor. In this case, the user is relying on her ISP (Verizon), Verizon routes all the payments through CitiBank, which in

turn clears through the national clearing houses, finally going to the Vancouver branch of Royal Bank with handles payments for the Vendor.

- In each path, each entity in the path is assigned the in-currency and the out-currency. That is, the Path Finder has already decided on who will do which conversion; this means each entity must specify to the Path Finder its charge for currency conversion (including the conversion rate, which should be periodically updated by each entity) and the Path Finder will pick the lowest total cost. This feature is necessary to keep possible paths down to a manageable level – otherwise the number of paths will explode (something exponential to the number of currencies involved).
- Alternatively, currency conversion is not part of the path. Each component specifies the conversion rate for conversions that it will performs. The user browser is then responsible for picking the currencies conversion as will. This further reduces the work of the Path-finder but increase the work for the browser.
- Each entity also must publish its charge for handling the transaction. The charge can be a fixed cost, say ½ cent USD, or a percentage, say 10%. In some cases (promotions, advertising, etc.) it could be -100% meaning user pays nothing and the advertiser pays the whole cost to the Vendor; or it could be even -100% - 5 cents which means not only is it free to the user, the user actually gets paid 5 cents!
- The User (actually the Browser/plugin) calculates the cost for a path by starting at the Vendor end with the desired price, go through the path hop by hop. At each hop, follow the published charge for that entity (currency conversion rate should be include in the path already).
- Some paths may be "funny" in that the user is paid money (see above) or it may be free but requires signing up to a mailing list or whatever.

6. User chooses which path to use
   - Present the choices to the User – this is to let the user select which path (implicitly selecting currency and Payer) and to confirm that the transaction should do ahead
   - This can be by a pop-up window, or it could be controlled by pre-configured settings like "All NY Times prices below 5 cents can go ahead", "request under 1 cent from the ISP package can go ahead" etc
   - This can be very simple or complex and is entirely under the control of the Browser program. So novice users will not have to understand the complexities while power users can have all the controls.
7. User asks Payer to certify this payment
   - After User picks a path, User has to send a request to her Payer to certify this payment.
   - The request will consist of the item-URL, the path, and User ID (or whatever the Payer needs to authenticate the user). This identifies what-is-bought, who-bought-it, who-will-pay, how-the-money-moves.
8. Payer sends Payment Certificate to User
   - The Payer performs its own business logic to decide if the User is authorized to spend this money, etc. If so, the Payer signs the transaction with the above information – this act certifies that the Payer will pay the amount to the next link in the chain. (If the Payer fails to pay, the PA up-the-hierarchy is responsible up to the big Clearing House)
9. User sends Payment Certificate to Vendor
   - After receiving the signed transaction, just send it to Vendor as part of the new HTTP get

request.

10. Vendor verifies Payment Certificate
    - Vendor receives the signed Payment Certificate and checks
        - the signature by the Payer
        - the certificate of the Payer has not expired.
        - In order to guarantee payment, we probably want to have a lot of limits on a Payer certificate – a combination of short expiry time, small limit per transaction, limit on total transactions per day per vendor. This limits the damage caused by a rogue PA since the PA can only defraud each vendor by a limited amount before action by the parent-PA.
        - Alternative security check is something like:
            - hash the payment certificate (this is to make sure the whole certificate contributes to the hash key)
            - some percentage of hash values (say all hash values divisible by 10) means the payment certificate should be confirmed with parent-PA (this is possibly part of Payer Certificate signed by parent-PA). The parent-PA is free to use any business logic to decide if the Payer has "turned rogue" or not. Some Vendors may decide to verify a higher percentage for high value transactions.
            - As a variation, it is possible to require the Payer to check the hash value and ask its parent-PA to sign the Payment Certificate. This makes it simpler on the user.
            - This has the advantage that only bad Payer needs to be tracked by vendor, and damage limit can be adjusted by the sampling percentage. Assuming average transaction is 1 penny, than sample rate of 0.001 mean each vendor will lose only $10 on average. Or a sampling rate 0.01 mean $1 on average.
11. Alternatively, the Vendor passes the Payment Certificate to the Vendor-Payer, and the Vendor-Payer does all this checking. This has the advantage that the Payment Certificate must be passed to the Vendor-Payer anyway, so it might as well do the checking – especially since it is responsible to collect.
12. Vendor sends desired item to User. At this point, the Vendor should be happy since payment has been guaranteed and can now do whatever magic to serve the content to the User.
13. User-Payer sends payment along path to CH
    - At some agreed upon time (weekly, daily, real-time, what-ever), payment is launched into the path by the first PA in the path (by definition, the PA that guaranteed the payment).
    - The first PA sends the signed transaction to the first CH in the path (possibly as a big batch of transaction all with the CH).
    - It should also do its own internal account to bill the User (and it could be free, part of a package, or whatever).
14. Clearing House clears
    - At agreed upon times, or enough received transaction, or whatever trigger is used, the Clearing Houses starts to clear the transactions.
    - The first step is for each CH to forward transactions to the next hop CH (there may be multiple CH in the path)

- After every CH has forward all its transactions, each CH has every transactions that include it.
- Each CH can now compute how money should move (and in what direction) between it and its peers.
- Each CH communicates with peers to confirm every agrees.
- If not, the two CH's can compare list of transactions. Since transactions are signed, it should be simple matter to resolve.
- If confirmed, money flow between CH.
    - The first and last CH in the path calculate the money for their PA and sub-PA (in both directions)
    - Net amount is moved

15. Receiver should have correct amount
    - The Receiver should receive the correct amount.
    - The CH or PA should be able to provide the list of transactions cleared.
    - The Receiver should be able to reconcile its own list against the provide list.
16. Everyone is happy!

Some performance statistics

- A 2048 bit RSA signature takes 6 milli-seconds on AMD Opteron 8354 at 2.2 GHz from 2009 (see http://www.cryptopp.com/benchmarks.html). We can safely assume that on 2013 processors, a signature will take a single milli-second. Note that verifying the signature takes much less work – assuming each entity will perform a signature is a very conservative estimate.
- For Amazon EC2, a "High-CPU Reserved Instance" is 36 cents per hour for "Extra Large" (for US East data center) which works out to 0.01 cent per second. Which means a RSA signature costs 10 micro-cents. This means we can over-provision CPU by a factor of 100 and it will still only cost us 1 milli-cent per signature.
- EC2 bandwidth is around 2 cents per GB (average of different plans and datacenters). Assuming each transaction will take 2K bytes:
    - 1000 bytes URL to identify the exact item being bought
    - 100 bytes of addressing and payment details for each entity
    - 10 entities in the transaction
    - The cost is then 500K transaction in a single GB, or 4 micro-cents per transaction each way
    - So, the cost to processing a single transaction is under 20 micro-cents for each of the entities in the chains. The total cost for all entities is well under ½ milli-cent.
- EC2 storage is around 25 cents per GB per month. Each transaction is 2 KB, so it will cost 12 micro-cents per transaction per month, or 0.3 milli-cent per transaction per year. Beyond that, archival storage should be even cheaper.

Processing load for the Payer

For the Payer Portal, one EC2 instance can handle 3.6 million requests per hour (60 minutes * 60 seconds * 1000 mill-seconds) and should be adequate for a user base of say million – depending on how many user are on-line and active at any time. Certainly a single CPU instance should be enough on a small to medium size ISP. As the software should easily parallelizable, more CPU instances can be turn on as needed.

A design for Micro-payment system for web contents

By Stanley Chow, March 4, 2013 All rights reserved

Updated May 27, 2013 All rights reserved

Updated October 9, 2013 All rights reserved

What business are we in

Enable the consumer to pay the content provider.

Secondarily, we also

Advertisers (and other third parties) to participate in the payment.

Our assumptions are simple: **Consumers want to pay for content**. More specifically:

- Many consumers browse contents from many sites. These sites are the result of hard work.
- Content providers want to get paid. This applies to the huge sites down to tiny sites. In many cases, the huge sites try pay-walls (and fail). The small sites don't even try pay-walls and just rely on advertising or consider it a labour-of-love.
- Content consumers want to pay and support content providers. This is especially true for small web sites since these sites are usually highly specialized and the content can be very difficult to find elsewhere. The consumer wants to support to the content provider.

What is our Content Business:

- Right now, consumers have no way to pay the providers. Google has just enter this market. This is good in that Google is validating the market. This is bad in that Google "is Google" and much bigger than us.
- We want to make it easy for consumer to pay. For example, if I do a search, and the top item is a NY Times article, I would look at the search summary and decide how much I am willing to pay. Let's say I am willing to pay my default of half-a-penny and NY Times agrees; then money is transferred and the article is displayed on my browser.
- We want the system to be flexible in terms of negotiating between/amongst different forms/avenues of payment.
- We want the system to be efficient in that we can process transaction as small as fraction-of-a-penny while the system has to be secure enough to handle transactions as large as thousands of dollars and aggregated payments of millions of dollars.
- We want the system to be multi-currency by design, since most web transaction will cross national boundaries.

Clearly, even though we are aiming at the web contents market, the system works equally well for eCommerce – we allow users to pay web sites in very flexible ways. The web site can choose to apply the money to content that is immediately displayed, or can choose to apply to money to physical objects to be shipped. We will describe the details in terms of contents and do market analysis only for the web content market.

What is our Advertising Business:

Given that each transaction is an expression of specific interest of the user, third parties are likely to

be interested. Examples:

- user looking up an article on a particular product – say a particular review on a new camera. The camera manufacturer may well want to pay for the cost of access.
- The manufacturer may well offer special deals like $50 coupons or free accessaries to make the camera more attractive.
- Other companies may want to offer deals related to the camera – bags, filters, memory cards, trips catering to photographers, etc.
- Researchers and survey companies may want to study certain group of users. E.g., the "Model Train Association of New York" may pay to reach people in New York who are accessing certain sites.

These "ad-inserts" could be inserted by any entities in the transaction, by using the payment negotiating machinery.

Market climate

Currently, in theory, content providers can get paid many ways:

- subscription
- per article
- advertising
- donation
- support operation (part of sales support, for example)

In practice, most content providers have only one revenue stream – advertising. This is problematic for many reasons:

- annoying to users
- advertising is controlled by ad agency, not content provider and there are increasing incidences of ads in bad-taste that are against the policy of the content site
- increasing a vehicle for malware/virus distribution – want to distribute a virus? Just buy an ad and you can have access to browser windows!
- less and less effective, countered by more and more plugins like adblock

Many content providers have tried many different ways to monetize their content:

- subscription – typically $20 per month, and most such efforts fail for lack of subscribers.
- Per article – typically $5 per month, up to thousands for specialized reports.

Content Market size (primary)

Our primary market is the everyday user doing everyday lookups, probably through search engines like Google. We can (conservatively) estimate the total transaction size:

- http://www.internetworldstats.com/stats.htm has internet user stats
- North America has roughly 275 million internet users. Assume only 1 in 10 users will be willing to pay US$ 5 per month for content. That means 27.5 million users * US$ 5 /month = US$135.5 million/month or almost US$ 1.65 billion/year.
- Europe has over 500 million internet users. Assume only 1 in 20 users spending US$ 3 per

month. That means 25 million users * US$ 3 /month = US$75 million/month or US$ 0.9 billion/year.

- Asia has over 1 billion users (mostly China, India and Japan). Assume 1 in 40 users spending US $2 per month. That means 25 million users * US$ 2 /month = US$ 50 million/month = US$ 0.6 billion/year
- Ignoring the rest of world (Africa, Middle East, Latin America, Australia), we get US$ 1.65 + 0.9 + 0.6 = US$ 3 billion/year. This is total amount of money paid for content. Say 80% goes to content providers and 20% (US$600 million/year) comes to "The Enablers".
- We have to split that money with many people (see later), but we should capture 10%, which means we make US$60 million/years; almost all profit in the form of royalty/licensing fees. Very Conservatively!!

Content Market size (Secondary)

We also aim at "professional" or specialized markets like Doctors, Lawyers. We can estimate the market size for Doctors:

- roughly 3 doctors per 1000 people. (The range is from 2 to 4 for the countries that we are counting)
- North America has roughly 90 thousand doctors. Assume 1 in 2 will be willing to pay US$ 50 per month for specialized content. That means 90 thousand doctors * US$ 50 /month = US$4.5 million/month or almost US$ 54 million/year.
- Europe has over 170 thousand doctors. Assume only 1 in 2 users spending US$ 50 per month. That means 170 thousand doctors * US$ 50 /month = US$8.75 million/month or US$ 105 million/year.
- Asia has over 300 thousand doctors (mostly China, India and Japan). Assume 1 in 3 doctors spending US $20 per month. That means 300 thousand doctors * US$ 20 /month = US$ 6 million/month = US$ 72 million/year
- Ignoring the rest of world (Africa, Middle East, Latin America, Australia), we get US$ 54 + 105 + 72 = US$ 230 million/year. This is total amount of money paid for "doctor" content.

There are many specialized fields: Doctors, Lawyers, Pharmacist, Engineers – of many different disciplines , Physicist, Mathematicians, Teachers – of different level and disciplines. It is easy to count dozens of fields. If we assume the Doctors are more of less typical, than the total size is US$ 230 million/year * a few dozen; which is easily a few billions per year!

The secondary market (total of many fields) is easily larger than the primary market!

Total market can easily reach US$10 billion /year, with our final net being $200 million /year very conservatively!!

Advertising Market size

There are many different segments of advertising that we can enter:

- Context-aware advertising – like Google Adsense. Google makes around ¼ of its revenue or $10 billions a year.
- Context-aware coupons – like Google Adsense, but with free offer to pay for content

- more "integrated" presentation in that ads are not just annoying pop-up, pop-unders, pre-trailers, etcs.
- More opportunities for multi-party advertising and special deals – not limited to a single ad-space seller. (This comes from our decentralized design with multiple competing entities).

How does our system work

To introduce our system, let's start by thinking about books. A book going from author to reader passes through many hands, each with a different role to play:

- author writes a book
- publisher – sometimes a single publisher, but could be multiple publishers (say one per country)
- advertising – the author goes on a talk show, publisher takes an ad in a magazine, etc.
- wholesale distributors
- retailers – bricks and mortars book stores
- buyer – perhaps as present for our reader
- reader

We could add other complications – library, eBooks, but this simple model is sufficient for now.

We will introduce out micro-payment system in a series of increasing complex scenarios. "The first set of "simplistic" examples show the process as seen by the user. The second set of "realistic" examples show more of the underlying process that allows the money to flow. We will than add the machineries that allow advertising.

A simplistic example of a simple case

We start with the simplest case of a consumer buying content from a web site:

- author (or content provider)
- publisher (the web site)
- advertising (the search engine)
- buyer/reader – we merge these roles for now

The detailed steps:

1. search engine (or some other way) points user to a page on a web site. (Note that, like advertising, the search result could have been a paid-for link on the results page, which is completely independent of the operation of paying for the content.)
2. User clicks on link, which means browser does HTTP get.
3. In absence of our system, the web site returns the page HTML (typically with a HTTP status code of 200 meaning OK). With our system, the web site returns a request for payment (with specific pricing, say 1 penny USD for the page). The request could be normal HTML with special tag marking it as a payment request, or the the request could be a new HTTP status code with associated payment information. There are many ways of doing this.
4. When Browser program receives request for payment, it pops up a window asking user if user is willing to pay that amount.
5. If yes, the payment is deducted from the User Wallet and a guarantee (probably in the form

of a cryptographically signed packet) is sent to web site (typically as another HTTP get, perhaps with a parameter for the payment).

6. Web site checks/accepts payment and returns the page HTML as normal.

For this scenario, it outwardly looks similar to most other payment systems. Most other payment systems seem to stop at this scenario.

A simplistic example of an ISP subscriber

We expect that most consumers will in fact not pay from their own cash. Nobody want to handle payments of pennies; indeed, there is no sensible way to handle payments of fractions of a penny.

Most consumers will in fact get bundled deals – either from their ISP (Internet Service Provider), or some other source. They may get to spend up to $5 on content as part of their monthly ISP fee.

- ISP allow $5 per subscriber per month, but clearly expect the average utilization to be much lower (in the same way that average bandwidth utilization is lower than maximum allowed).
- User has $5 credit (for this month) in her wallet, but this is only credit from ISP, not actual cash.

The detailed steps:

1. as before, search engine points user to a page on a web site.
2. As before, User clicks on link, which means browser does HTTP get.
3. As before the web site returns a request for payment (say 1 penny USD for the page).
4. As before, when Browser program get request for payment, it pops up a window asking user if user is willing to pay that amount.
5. New for this scenario. If yes, the Browser confirms that the ISP is willing to guarantee this payment, and forward the guarantee to the web site. There are many ways to do this – the simplest way is to just forward the payment request to the ISP and receive the guarantee which is then forwarding to the web site. Essentially, the web site is receiving payment from ISP, but sending content to user.
6. Web site checks/accepts payment and returns the page HTML as normal.

This scenario is beyond what the competitions can do.

A simplistic example of a professional

For many fields, there are professional associations and other entities that will want to "aggregate" information content. For example, CPA (Certified Professional Accountants) are at the top of the accounting profession in the USA. The AICPA (American Institute of Certified Public Accountants) does the normal stuff as a professional association, including keeping the members informed about news related to accounting. The AICPA could add a member benefit for access to specialized web contents including some number of journals, newspapers, etc. The AICPA would give each member a credit much like the ISP gives credits to subscribers.

Many consumers will have credits from their ISP as well as one or more professional/specialized associations. This makes the process a little more complicated:

1. as before, search engine points user to a page on a web site.
2. As before, User clicks on link, which means browser does HTTP get.
3. New for this scenario, the web site returns a request for payment (say 1 penny USD for the page). The request will also contain a list of associations for which there is a special deal

and the details of each deal. The deal could be free unlimited access for members of the associations, or it could be half price for pages, etc.

4. New for this scenario, when Browser program get request for payment, it matches the different deals against the wallet and see which deals are applicable. The browser then pops up a window asking user if user is pick which deal, or confirm willingness to pay that amount.
5. New for this scenario. After user selects the deal, the Browser confirms that the ISP/association is willing to guarantee this payment, and forward the guarantee to the web site.
6. Web site checks/accepts payment and returns the page HTML as normal.

This scenario is well beyond what the competitions can do.

How does money flow?

In the simplistic example of a simple case, we had only the user and the web site. How does that penny move from user to web site? Well, in the early days of eBay, users literally tapes coins to post cards and mailed in the money. This is really not efficient due to the cost of post card and the cost of postage. This is also not going work well across national boundaries.

We look at how credit cards work and use that as a model.

In the credit card model, the buyer and the merchant each deals with a bank. The buyer signs a sales slip that promises to pay the transaction amount to the bank. The money moves from the buyer to the buyer's bank to the merchant's bank to the merchant. (Of course, at each step, many transactions are netted out and totaled, so that the user pays one monthly bill, the banks net out all account and only pay each other the difference). In fact, the banks do not deal directly with each other, they each deal with a clearing house.

It is also instructive to look at how X.509 certificates work. A CA (Certificate Authority) is a trusted third party that can issue certificates. If you trust that CA, then you trust the certificates issued by that authority. There are also mechanisms for sub-CA and sub-sub-CA like a tree. Each CA (sub- or otherwise) can revoke any certificate issued by it.

When we adapt the credit card model and the X.509 model to our system, we come up with the following roles:

- "user" - the entity who is doing the web browsing and consuming the content. Think of the boyfriend who wants a beer.
- "Payer" - the entity who actually pays. This is how money enters our system. This can be individual users, but is more typically an ISP or association paying in bulk. Think of the nice girlfriend buying a beer for her boyfriend with a credit card (and we give her a X.509-like certificate to sign her purchases).
- "Receiver" - the entity who actually receives the money. This is how money exits our system. Typically, a web site providing content. Think of the beer store accepting a credit card purchase (and we also it a X.509-like certificate for ease of some authentication protocols)
- "Payment Authority" - analogous to the CA, a PA guarantees payments. The PA can sign/guarantee a sub-PA and sub-sub-PA in turn, eventually signing/guaranteeing the Payer. Think of the PA as MasterCard, the international association. Note that there can be multiple PA competing, just like Visa, Amex competes with MasterCard.
- "sub-PA" - Think of the sub-PA as the bank that issues the credit card. There is likely to be 2

sub-PA involved in each transaction – the first sub-PA is the one that issued the credit card to the Payer (and will handle payments from the Payer) and the second sub-PA is the one that will collect payments for the Receiver.

- "Clearing House" - This serves a similar role to the credit card clearing house. It is likely that there will be multiple Clearing Houses. We may require that all Clearing Houses will work together to clear transactions (in a way, all the Clearing Houses will form a Big Clearing House); this requirement may be relaxed if we allow some transactions to be not completable since there is no path from the paying sub-PA to the receiving sub-PA (this will introduce complications in all handling of transactions, so it is recommended that all Clearing Houses must work together). Note that not completable paths will still occur due to other things – not having a specific kind of credit (e.g., I don't belong to AICPA, or I have already spent all my AICPA credits), or no clearing house will do the currency exchange (may be one currency is officially not convertible.)

In the same way that people can choose which credit cards they carry, each Payer can choose which PA or sub-PA she deals with. Each Receiver also has freedom to choose with PA or sub-PA.

A realistic example of an ISP subscriber

Recall the this example has an ISP subscriber spending credits from ISP.

When the Payer Browser receives request for payment, just check that there is enough credit in the wallet.

A realistic example of an professional

Recall that this example has a professional spending credits from her ISP as well as credit from her professional association.

The web site sends a request for payment, the Payer Browser will check each credit in the wallet to see which credits has a path to the Receiver sub-PA. The user is then asked to pick the credit to use.

How do we make money

The previous section enables the Payer to pay the Receiver. This section discuses how each player gets its share of the money.

When the content provider sends out a request for payment, the request contains:

- the amount of money wanted, possibly different amounts in different currencies. Currencies certainly includes national currencies, but can also be many other forms of value – loyalty program points, frequent flyer miles, bitcoin, membership benefit credits, etc.
- List of sub-PA that is acceptable to the Receiver – probably the list of sub-PA with whom the Receiver has a contract.
- Optionally, a blob of "routing optimization" data. This is to make it easier/faster/simpler for the "Path-finder".

When the Payer receives this request, the Payer's own list of sub-PA is processed with the Receiver's list. The processing is to find all paths form Payer's sub-PA, through PA and Clearing Houses, to the Receiver's sub-PA. If, as expected, there are many thousands of PA's, millions of sub-PA, it can be time consuming to general all paths; so we expect each end to pre-compute and cache some data. It is also likely that the PA and/or sub-PA will perform the "Path-finder" function instead of relying on the Payer.

After all paths are generated, each path is evaluated. Each node in the path will get a "fee" for

handling the transaction. The fee for each node is specified by the node – like Visa and MasterCard each can specify how much they want for processing a credit purchase. In our model, the request for payment contain the amount of money wanted by the Receiver, we can work backward from that to see how much the payer has to actually pay. The viable paths can be presented to the user for selection (or the user can set a default that the cheapest path is always used). Note that it is up to each node to decide its fee; but if its fee is high, then users will simply not select any path containing it. Thus we encourage price competition.

There can be multi-currency and different package deals. For example, a request for payment could say:

- 15 pennies in USD, must clear through sub-PA-US
- or 100 Yen JPY, must clear through sub-PA-JP
- or 10,000 AAmiles, must clear through sub-PA-AmericanAirlines
- or 2 credits from AICPA, must clear through sub-PA-AICPA

The Payer will look at whether she can get the 2 AICPA credits, and the total cost in USD as the Payer needs to pay (including all transaction fees and currency exchanges). The Payer/user can then pick which path – the choice of whether to use the AICPA credit may depend on the user wanting some AICPA-exclusive content in the future or perhaps it is the end of the month and she will get some more AICPA credits tomorrow.

Technical note: to speed up the generation of paths, it is likely that each node should specify specific properties – for example, a node can declare that it satisfies the "triangle inequality" - that is, going from the node to a neighbor will always be cheaper then going through intermediate clearing houses.

Interesting aspects of this system

This system has some interesting properties that are necessary for success:

1. System is "consumer aggregated" in the sense that consumers knows whom to pay and how much to pay. It is a Single Point of Payment as opposed to paying each provider separately. This is a prerequisite for efficient operation of the system, and is a prerequisite for User-Friendliness. In other words, I am willing to pay $4.23 for this month, but I am not willing to pay hundreds of bills each of 1 penny or two.
    - User can pay sub-PA in a net bill (like a credit card bill)
    - User can also get credits from ISP or other associations, possibly as part of bundles.
2. System is "provider aggregated" in the sense that providers knows from whom to collect and how much to collect. Single Point of Payment is necessary for the same reasons above. The web site does not want to collect pennies, instead, the web site want to have someone else collect all the pennies and just give a single cheque to the web site.
3. System has audit trail. Even though the transactions are tiny, the aggregated payments must be resolvable to each transaction, and each transaction must authenticate to each payer/payer with complete traceability.
4. System is not anonymous. Even though this system targets tiny transactions, there is nothing limiting the size or the number transactions; so aggregate size may be substantial. This mean there is potential for money laundering, so each transaction must be traceable to payer and payee. This may be "against the grain" for many researchers into payment systems, but we must avoid the this large legal minefield. Since all payments are done through PA/sub-PA,

we can identify Payer and Receiver to the sub-PA. It is then up to the sub-PA to satisfy local laws and customs.

5. Since transactions will typically happen first, then money move later (by hours or days), we guarantee that money will move. Again, each individual transaction may be small, aggregates will be large (if that wasn't the case, we wouldn't be interesting in this business). It is up to each PA to guarantee its sub-PA. This nay translate to PA and sub-PA posting performance bonds, and a more complex design in the traceability and/or authorization of each transaction.
6. By design of the system, it is expected that there will be competition at every level (except as the Clearing House level) and the competition is not just that there are competitors but each transaction is a mini-competitive bidding selection process.
7. Each user can have many different credits in her wallet – one from ISP, one for general news, one for professional interest, one for a hobby, … This system makes it easy to handle many credits and still take optimal advantage of the different attributes of each credit.
8. By combine the traditional retail model with the credit card model and the Certificate Authority model, we have come up with an unique design that works well, scales to world-wide deployment and yet meets all requirements.
9. Deployment and market acceptance can be gradual. Each jurisdiction can decide if it want competition – if yes, just allow multiple PA's. If not, just allow a single PA. It can join the Clearing Houses that it wants. There is no need to have the whole world convert at the same time.

Example usage with advertising

Advertising can be injecting into a transaction by different entities. For example:

- the content provider can specify the price as "free with advertising" which is basically what the current advertising-based web sites do, so this case is not very interesting for us.
- The payer sub-PA-AICPA could insert advertising. For example, BMW USA want to target CPA's (presumably likely buyers of high-end cars like BMW), but not just any CPA; the company would like to target CPA's looking for information on reviews of cars. Currently, there is no way to do this at all. With our system, BMW USA could contract with AICPA so that when the sub-PA-AICPA "sees" a transaction for the "right" article, the net cost is calculated to be free (or even a special offer). This special offer could be even be conditioned by the membership grade and other information. It is possible that for new CPA's the special offer is for a series 3 BMW, while for long time members (or senior members) the special offer will be for a series 7 BMW. The possibilities are endless.
- Just about any other entity can also inject advertising, the only limitation is what the entity know. Note that entities can accumulate information. For example, the sub-PA-ISP may observe that the reader is particularly interested in certainly subject and insert advertising accordingly.
- This (potentially) allows the ISP (and other providers of credit) to build up a comprehensive profile of the users. This profile is much more complete than the profiles from loyalty cards, etc. This may be attractive enough for data aggregators to get into this market.

Competitive analysis

Most of the past and current attempts have been "Centralized" systems with a few "P2P" systems. By Centralized, we mean one company controls the entire system – an excellent current example is Google micro-payment. By P2P, we mean systems like Bitcoins where there is no central authority that controls the system. We will address the weaknesses of each kind in turn after

The P2P systems are doomed since it fails essentially every point above.

The Centralized systems are more

Current Competitors

1. http://boingboing.net/2011/11/11/this-28-year-olds-startup-is.html Company is Dwolla, a flat fee of $0.25 per transaction. The average transaction volume for Dwolla is right around $500 dollars. We move between $30 and $50 million per month.
2. http://www.cbc.ca/news/technology/story/2012/04/12/technology-mintchip-digital-penny-payment.html Digital money competition by Royal Canadian Mint. Winner selected.
3. http://bitcoin.org/ Completely open P2P currency. You create your own money. No authority – central or otherwise. Currently in use.
4. http://www.paypalobjects.com/IntegrationCenter/ic_micropayments.html Paypal's idea of "micro" is charging "a special rate of 5% + $0.05 per transaction" which is more than our idea of transaction.
5. http://www.techspot.com/news/51788-ea-micro-transactions-will-be-built-into-all-of-our-games.html Electronic Arts is building micro-transactions into their games.
6. http://www.qidian.com/Default.aspx (in Chinese) Typical Chinese site that sells serialized novels, accepts money in form of pay-as-you-go cards for phones as well as games. Typical content sells for small number of penny (USD), even fractions of a penny. Closest to what we aim to do.
7. https://gumroad.com/ Not really micro-payment as much as an eCommerce automation service. collecting a 5% transaction fee plus 25 cents per each sale. More like Paypal kind of micro.
8. Many content providers (NY Times, Wall Street Journal, …) have tried different kinds of Pay-walls – limited number of free articles per month, only old articles are free, etc.
9. http://www.digitaltrends.com/web/google-launches-micropayments-for-web-content/ Google is aiming squarely at our target market, but with Google as the central hub for everything. This is probably our most serious competitor. Google will have lots of problems gaining traction since it needs its competitors to adopt its micro-payment system. This is unlikely to go down well.

Old competitors

- http://sellitontheweb.com/blog/millicent-micropayment-product-review/ Old DEC effort in 1998. Buy "scrips" from vendor.

- http://techcrunch.com/2011/05/31/minno-rebrands-as-buysimple-partners-with-soundcloud-raises-700k-for-micropayments/ It is no more. Just one of many failed attempts.
- http://en.wikipedia.org/wiki/Peppercoin Invented by Ron Rivest (the R in RSA), company start in 2001 buy sold to Chockstone in 2007. It is no more.
- http://en.wikipedia.org/wiki/Micropayment Wikipedia entry listing many system including systems by IBM, DEC starting in 1998.