Blockchains Architecture, Design and Use Cases

Prof. Sandip Chakraborty
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 11
Consensus in Bitcoin – I (The Basics)

Welcome back to the course on Blockchain.

(Refer Slide Time: 00:22)

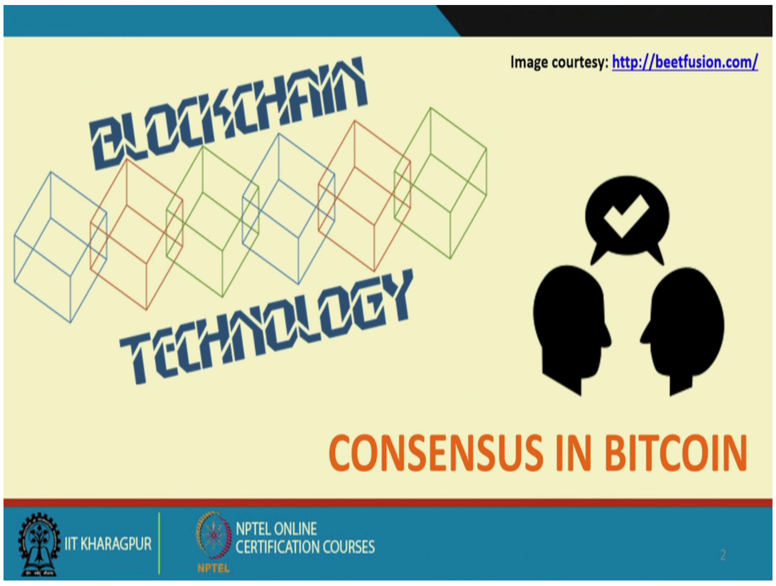
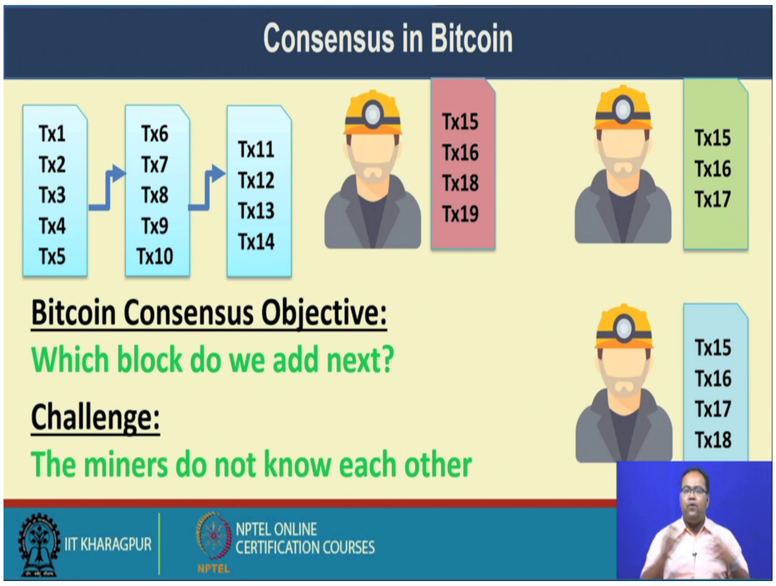


Image courtesy: http://beetfusion.com/
BLOCKCHAIN
TECHNOLOGY
CONSENSUS IN BITCOIN
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL
2

So, in the last discussion we have looked into the consensus mechanism in blockchain, consensus are the general distributed consensus architecture. So, in this particular talk will look into that how consensus is achieved in bitcoin based permission less blockchain environment.

(Refer Slide Time: 00:40)



Consensus in Bitcoin

Bitcoin Consensus Objective:

Which block do we add next?

Challenge:

The miners do not know each other

So, here is the consensus algorithm in bitcoin. So, the objective of the consensus algorithm is in bitcoin is to add a new block to the existing blockchain.

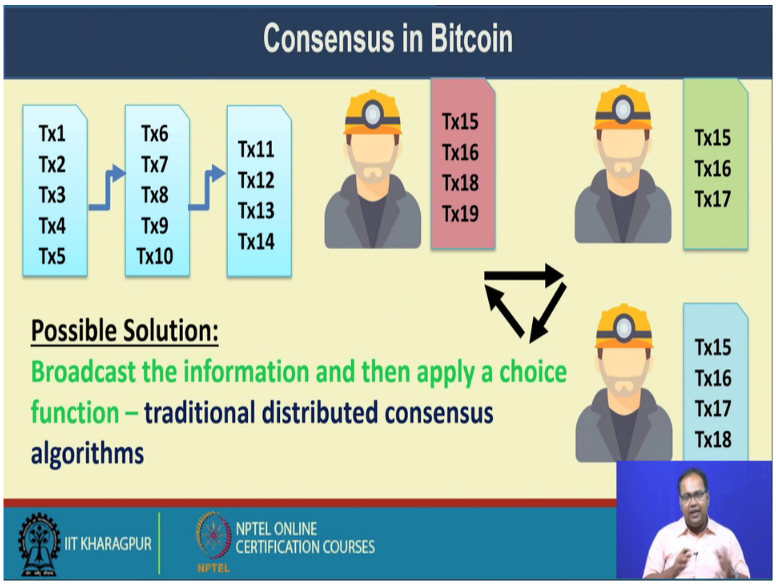
Now, there can be multiple minors in the bitcoin network and individual minors can propose their new blocks based on the transaction that they have part off. So, one minor can had the transactions 15, 16, 18 and 19, whereas another minor can had the transactions 15, 16 and 17.

So, it is interesting to note as I have also mentioned in the last lecture that it is not necessary that every minor will propose a new block or better to say every minor will propose a same block. So, it is it is all right that whatever transactions the minor has hard of till now, or better to say that the time that the last block has been added in the blockchain they can include all those transactions in the new block provided that the total size of the block does not exists certain threshold.

So, in this particular example will have 3 minor. So, this individual 3 minors have the 3 individual blocks which have been proposed by them and objective here is to decide that which particular block need to be added to the existing blockchain. So, the consensus objective is that which block do, we add next out of the 3 blocks proposed by the minor.

Now, the challenge here is that the minor do not know each other, because it is a open network, it is a permission less network. So, anyone can participate in the network as a minor and they can collect the transactions coming from the clients and they can propose a new block. So, under this certain scenarios how will you ensure that the minors will come to our consensus?

(Refer Slide Time: 02:44)



Consensus in Bitcoin

Tx1
Tx2
Tx3
Tx4
Tx5
Tx6
Tx7
Tx8
Tx9
Tx10
Tx11
Tx12
Tx13
Tx14
Tx15
Tx16
Tx18
Tx19
Tx15
Tx16
Tx17
Tx15
Tx16
Tx17
Tx18

Possible Solution:

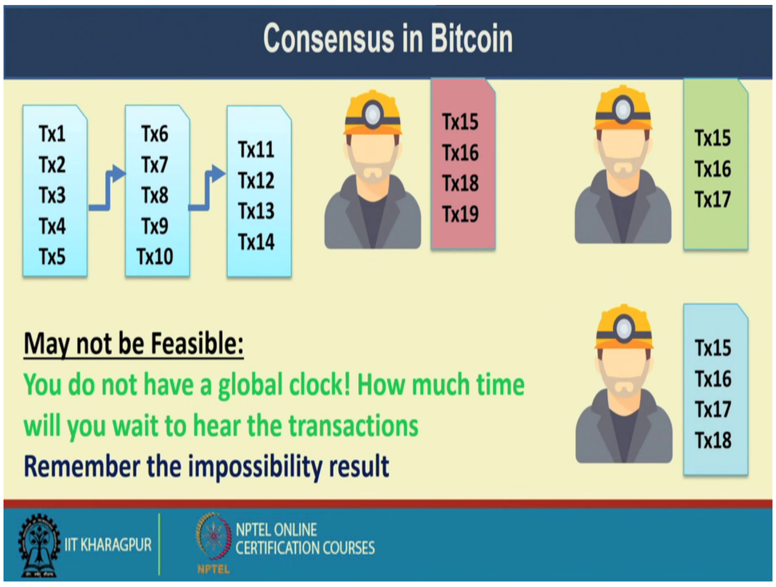
Broadcast the information and then apply a choice function – traditional distributed consensus algorithms

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, let us look into one possible solution which is more closer to the traditional distributed consensus algorithms, where every minor whenever they are proposing some new blocks they will broadcast the information in the network and after broadcasting the information to the network they will wait for a certain amount of time and see that whether they are disturbing any other blocks from other minors and based on that they will apply a choice function to select one of the blocks, but there is a problem in this particular mechanisms. So, what is the problem? The problem here is that whenever we are saying that a minor will wait for certain duration that is technically infeasible under this particular environment.

Why? Because first of all the miners do not know each other because the minors do not know they do not know that how many other minors are there in the network who are going to propose new blocks. And the second to one is that the internet is inherently asynchronous, asynchronous in the same slide whenever a minor will propose a block the there can be arbitrary delay in transmitting those new blocks.

(Refer Slide Time: 04:01)



Consensus in Bitcoin

May not be Feasible:

You do not have a global clock! How much time will you wait to hear the transactions

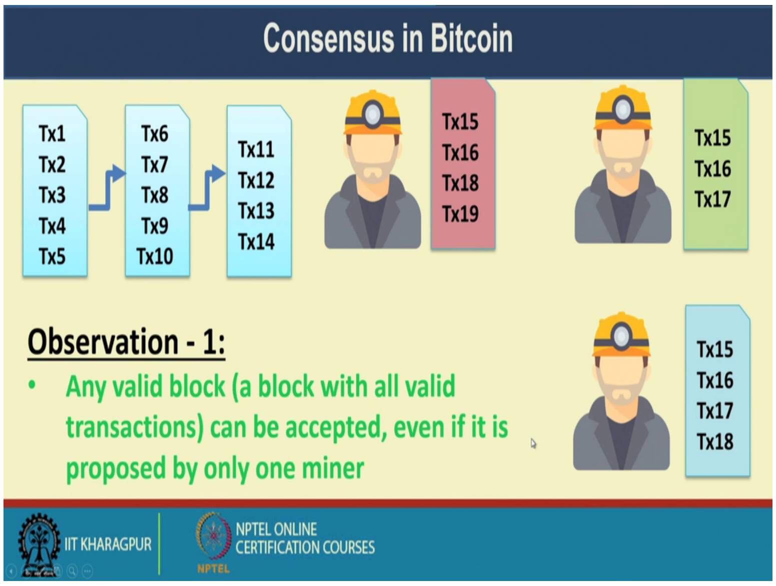
Remember the impossibility result

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, we in this particular environment having a consensus based on a broadcasting algorithm by applying traditional distributed system mechanism may not be feasible. So, if you remember the impossibility theorem where we have mentioned that coming to a consensus is impossible in the in a purely asynchronous distributed network in the presence of even a single failure.

So, in this particular architecture even if there is an minor which is an malicious minor, even in the presence of a single malicious minor other minors may not be able to collect all the blocks which are coming from the legitimate minors and having a consensus by applying this kind of traditional distributed algorithm based on broadcasting or based on message passing is infeasible in this scenario.

(Refer Slide Time: 04:57)



Consensus in Bitcoin

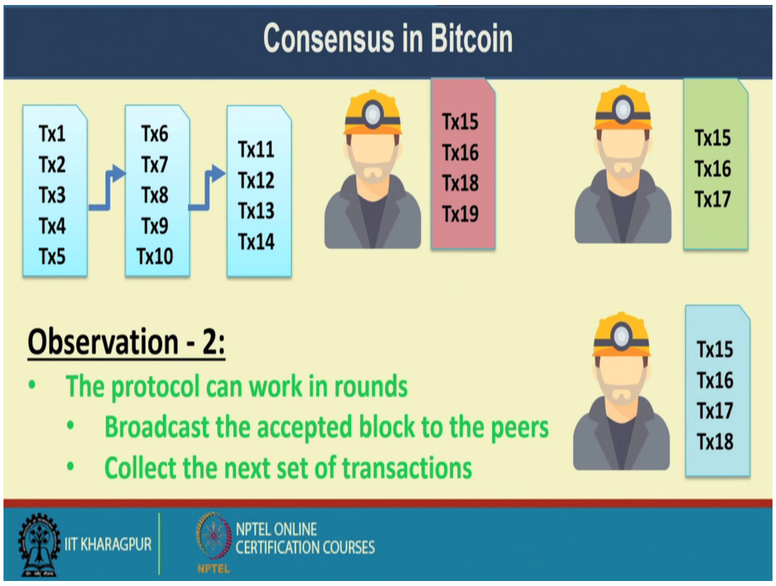
Observation - 1:

- Any valid block (a block with all valid transactions) can be accepted, even if it is proposed by only one miner

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, let us see that what we can do. So, we have two different observations here. The first observation is interesting that in case of a blockchain environment any valid block can be accepted. So, a valid block means a block with all valid transaction. So, we can accept any of the proposed block if it is valid. And even if the block is proposed by a only single one minor. So, it is not necessary that the block need to be proposed by multiple minor together even if it is coming from a single minor and it is a valid block we can accept that block and we can connect that block to the existing blockchain.

(Refer Slide Time: 05:38)



Consensus in Bitcoin

Observation - 2:

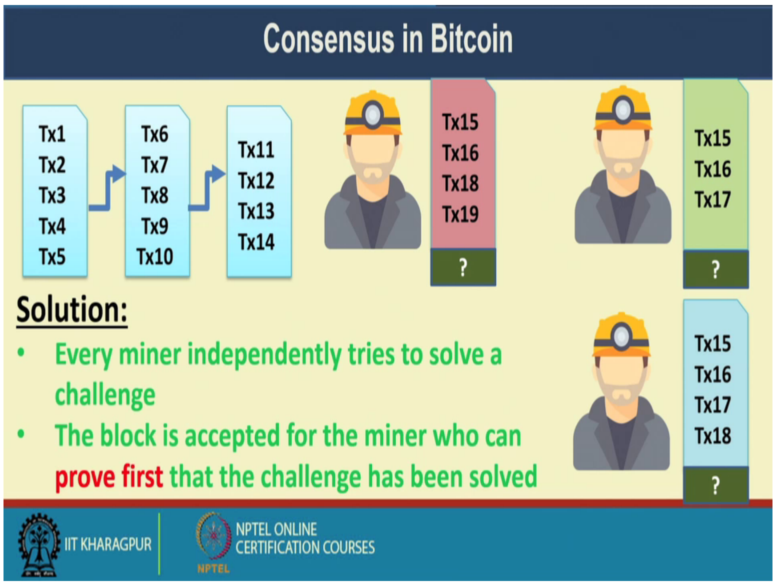
- The protocol can work in rounds
  - Broadcast the accepted block to the peers
  - Collect the next set of transactions

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, that was our first observation. And our second observation is start the inter protocol can run in rounds run in rounds in means like once a valid block has been accepted or stay the system has reached to the consensus from that point round starts the minor starts getting the transactions, and then they can find out that which are the transactions which are not already committed in the blockchain and based on that they can propose the new block.

So, if you look into this particular example in this example the last block in the blockchain it has transaction 11, 12, 13 and 14. So, all the minors wants this particular block has been committed in the blockchain the minors they can find out that well transaction 11, 12, 13 and 14 has already been committed. So, we do not need to include those transactions again in a new block. So, they can propose a new block by excluding the transactions which has already committed in the blockchain. So, that way once the commitment has been done that means, the block is added to a particular blockchain the minors can try to figure out which are the next transactions that need to be added to the blockchain and they can proposed the new block based on that.

(Refer Slide Time: 06:56)



Consensus in Bitcoin

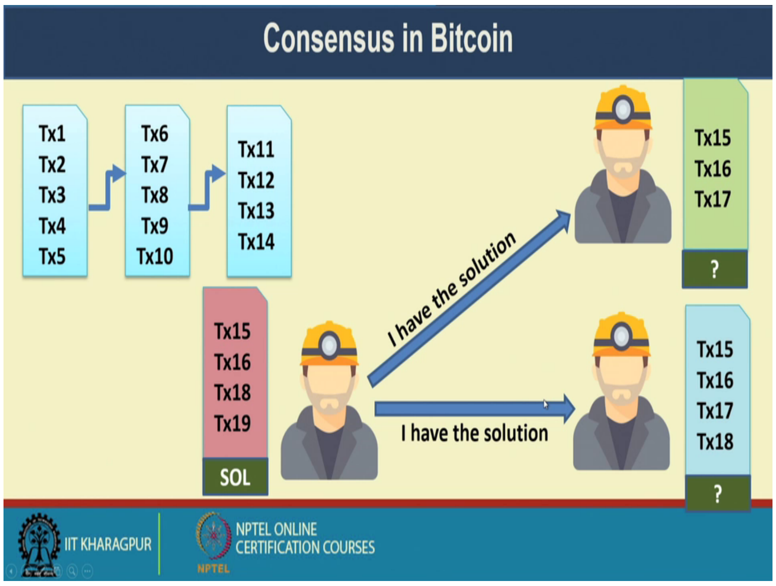
Solution:

- Every miner independently tries to solve a challenge
- The block is accepted for the miner who can prove first that the challenge has been solved

So, based on these two observations we can design a solution in this way where every minor they independently tries to solve a challenge. So, the challenge is provided by the network. So, they will independently try to solve a challenge and the block is accepted for the minor who can proof fast that the challenge has been solved.

So, this particular q r proof fast is important here because this q r says that every individual minors they are trying to solve a solution imposed by the network and the minor who is able to solve the problem first they at the solution of the problem to the existing block and add that block to the blockchain.

(Refer Slide Time: 07:44)



Consensus in Bitcoin

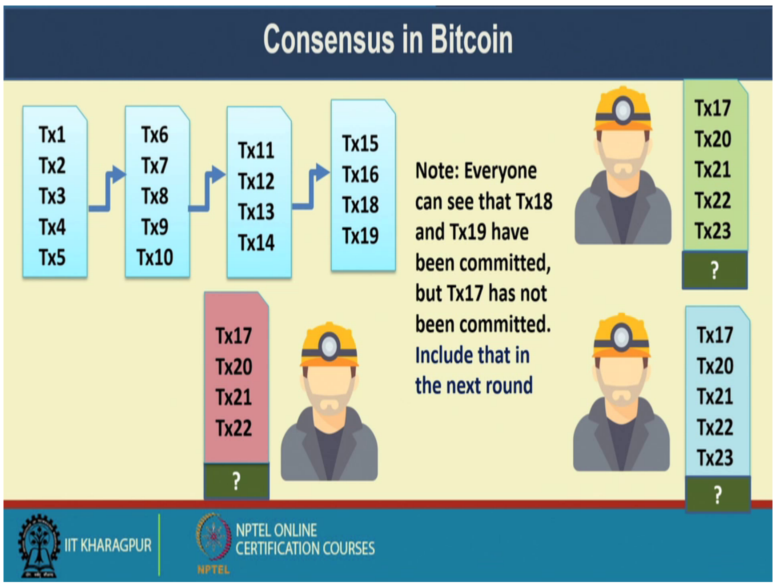
So, this particular algorithm is known as proof of work that proof comes from the fact that the minor he has been able to the minor who is proposing the solution he was able to proof that he has found out a solution of the problem. So, that is the solution architecture that when the minor finds out that there is a solution he proposed the block to the other minor saying that well I have already received the solution. So, he has already received the solution and other minor they stop finding out the solution for the challenge and start finding new blocks in the from the transactions that which are coming from the clients.

Now, one interesting point to note that this particular communication can work asynchronously. So, whenever you are advertising the solution to the other minors in the network, the minors can work for mining the previous block during that time say I am a minor I am trying to find out the solution for a given challenge and whenever I will find out that well a new block has been added in the blockchain.

During that time if I find out that well that particular block has certain set of transactions which are also part of my block. So, there is over lapping of transactions between the block that I am trying to proof or I am trying to mine and the newly added block in the blockchain. If that is the scenario then I can just stop the mining at that point accept the most updated blockchain and then start finding out a blocks with new transactions which are coming from the clients.

So, once the solution has been obtained. So, the block has been added to the existing blockchain. So, this block with transaction 15, 16, 18 and 19 has been added to the blockchain.

(Refer Slide Time: 09:42)


Consensus in Bitcoin

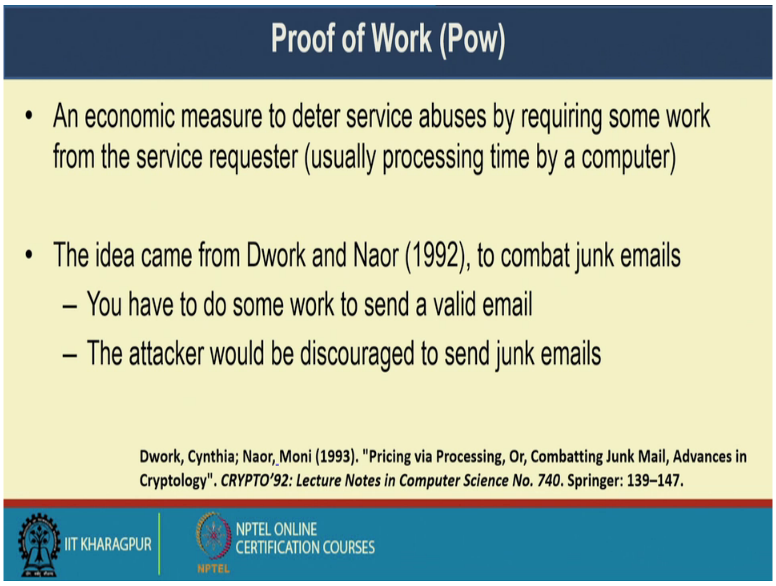
Note: Everyone can see that Tx18 and Tx19 have been committed, but Tx17 has not been committed. Include that in the next round

Now, the interesting fact that whenever this particular information this updated blockchain is propagated to all the minors in the network. So, all the minors can see. So, the other minors they were at that time they were also trying to mine the blocks where this transactions 18 and 19 were there. So, if you look into the earlier slide. So, this minor was also trying to mine a block where transaction 15, 16, 17 was they are the second minor was trying to mine a block where transaction 15, 16, 17, 18 was there.

Now, this new block has been added to the blockchain and they can find out that well in the newly added block transactions this 18 and 19; 15, 16, 18 and 19 they are there, but transaction 17 has not been included in the block. Now, in the next round every minor can include transaction 17 and at the same time along with transaction 17 the other transactions that they have hard within that time duration. So, those transactions can be added and they can start mining those specific blocks. So, the same process repeats in rounds and they can try to find out that what can be the solution for that particular challenge.

So, what would be the challenge? So, to generate the challenge bitcoin relies on a mechanism called proof of work.

(Refer Slide Time: 11:07)



Proof of Work (Pow)

- An economic measure to deter service abuses by requiring some work from the service requester (usually processing time by a computer)
- The idea came from Dwork and Naor (1992), to combat junk emails
  - You have to do some work to send a valid email
  - The attacker would be discouraged to send junk emails

Dwork, Cynthia; Naor, Moni (1993). "Pricing via Processing, Or, Combatting Junk Mail, Advances in Cryptology". *CRYPTO'92: Lecture Notes in Computer Science No. 740*. Springer: 139–147.

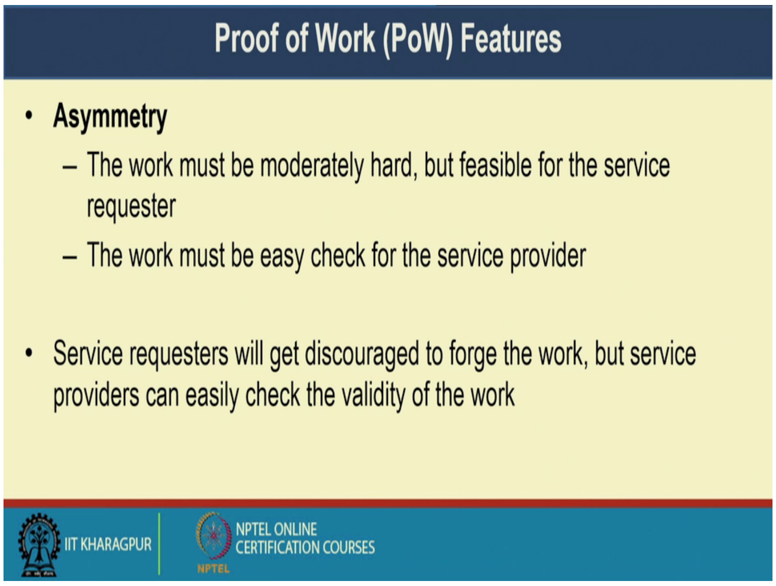
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, what is the proof of work? The proof of work is an economic measure to deter service abuses by requiring some work from the service requester. Usually it is processing time of a computer, say if you are requesting for a service you have to show to the service provider that you have spent significant interest on the service that you are requesting for.

So, I will give you an example that how this kind of proof of work based system can help you to prevent junk emails or spam emails that is an interesting idea, but the broad idea of this proof of work is that every service requester must spent some time on the service that he or she is asking for before actually requesting for the service. So, that way it shows you from the economic perspective that the service requester has some interest on the service. So, if the services provided to that service requester that service requester will genuinely accept that service and it will not use it maliciously.

So, this idea is actually an old idea, which came from Dwork and Naor in 1992 from an interesting paper published in Crypto Conference to combat the junk emails. So, in that particular paper the idea that they had shared that, you have to do some work to send a valid email. Now, the attacker this way they will be discouraged to send junk email because in that case they have to do the same work of the same complexity over the junk emails that will not proof beneficial for them.

(Refer Slide Time: 12:54)



Proof of Work (PoW) Features

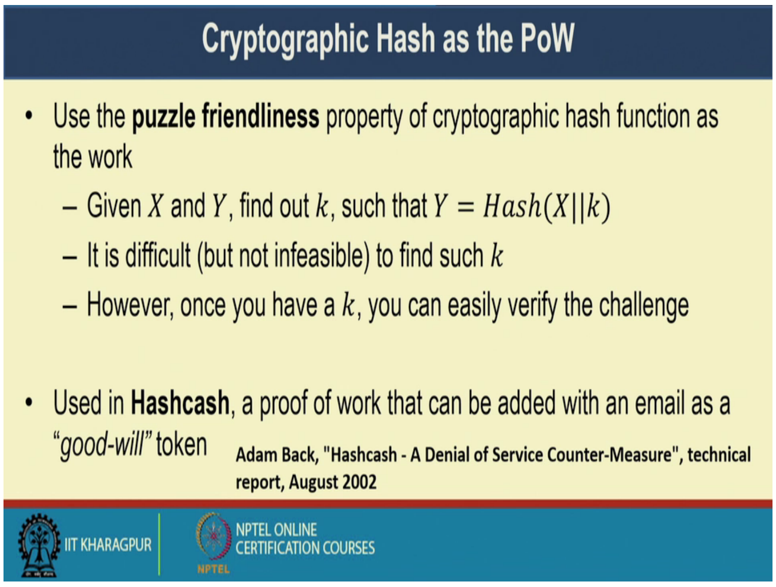
- **Asymmetry**
  - The work must be moderately hard, but feasible for the service requester
  - The work must be easy check for the service provider
- Service requesters will get discouraged to forge the work, but service providers can easily check the validity of the work

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, this kind of proof of work base system they should have a certain features one important feature is the asymmetry, that means, the work must be moderately hard, but feasible.

So, you should try to solve a moderately hard, but feasible work to the service requester side whereas at the service provider side the work must be easy to check or must be easy to validate. So, that way the service requesters they will get discourage to forge the work. So, because they have to spend some significant amount of time to complete that particular work they will be discouraged to submit any kind of spam or junk work to the system, but service providers on the other hand they can easily check the validity of the work because of the asymmetry nature that the work is easier to validate for the service provider, but it is hard, but not infeasible for the service requester.

(Refer Slide Time: 13:55)



Cryptographic Hash as the PoW

- Use the **puzzle friendliness** property of cryptographic hash function as the work
  - Given X and Y, find out k, such that $Y = Hash(X||k)$
  - It is difficult (but not infeasible) to find such k
  - However, once you have a k, you can easily verify the challenge
- Used in **Hashcash**, a proof of work that can be added with an email as a *"good-will"* token **Adam Back, "Hashcash - A Denial of Service Counter-Measure", technical report, August 2002**

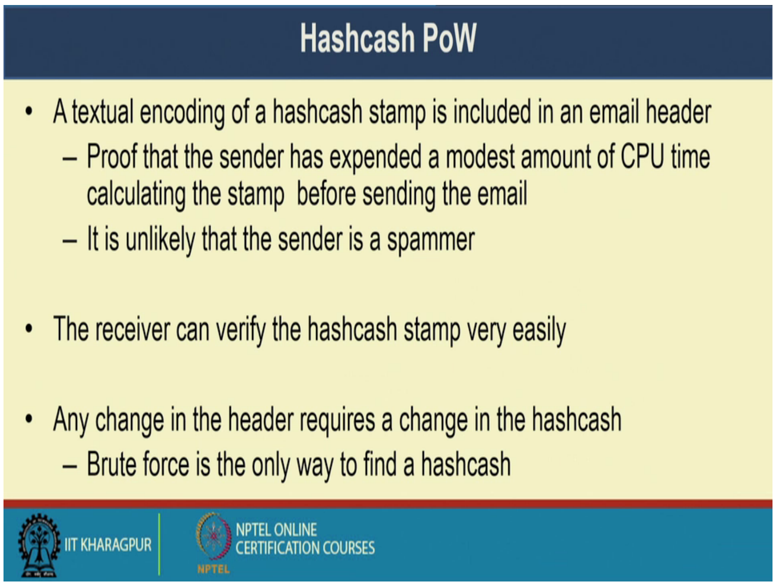
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, let us look into that how cryptographic has can work as a good indicator for proof of work. So, while discussing about different type of cryptographic hash functions. We have looked into the puzzle friendliness property of cryptographic hash functions. So, the puzzle friendliness property says that even X and Y, you need to find out k such that Y is equal to hash of X and k appended with each other. So, if X and Y are known in that case it is very difficult to find out the certain k you have to the best algorithm to find out this kind of k is to apply a certain kind of route force mechanism or some kind of random search a over the possible values.

So, this particular work it is difficult to find, but it is easier to validate. So, once you have find out such k, it is easier to find out or it is easier to validate whether the hash value of X appended k becomes equal to Y or not. Now, this concept was used in hashcash.

So, this hashcash work was proposed by a Adam back in 2002, which was a proof of work mechanism that can be added with an email as a good will token. So, it is like that the hashcash can be only generated or the legitimate email senders will be interested to generate a hashcash whereas, the spam email generators they will be discouraged to generate the hashcash. So, let us look into the details of this hashcash mechanism and how it works.

(Refer Slide Time: 15:40)



Hashcash PoW

- A textual encoding of a hashcash stamp is included in an email header
  - Proof that the sender has expended a modest amount of CPU time calculating the stamp before sending the email
  - It is unlikely that the sender is a spammer
- The receiver can verify the hashcash stamp very easily
- Any change in the header requires a change in the hashcash
  - Brute force is the only way to find a hashcash

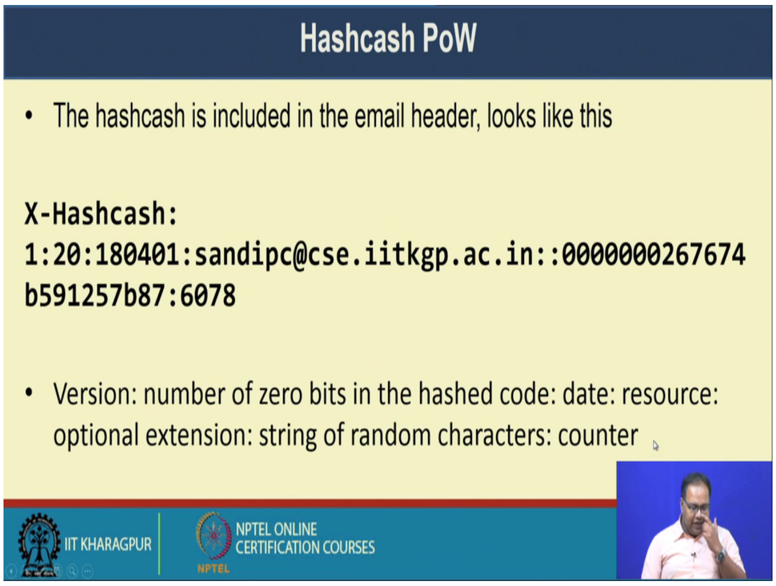
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

Now, this hashcash proof of work is an textual encoding or a you can think of it as a hashcash stamp which is included in an email header. So, this hashcash is a proof that the sender has expanded a modest amount of CPU time to calculate the stamp before sending that email.

Now, if the sender has spend a modest amount of time to generate the hashcash before sending the email it is unlikely that the sender is a spammer because the spammer will be interested to send repeated emails one after another by some auto generated script. So, if they have to wait for certain amount of time for sending every emails that the spammer will not be beneficial out of that. But on the other hand the email receiver they can verify the hashcash spam very easily by applying certain hash function.

But any change in the header also requires a change in the hashcash because you are generating the hashcash from the header. So, if you want to modify certain field inside the email header you have to re-generate the hashcash which is for which the brute force is the only way to find a new hashcash. So, that way attacker will also be discouraged to tamper the email header.

(Refer Slide Time: 17:01)



Hashcash PoW

- The hashcash is included in the email header, looks like this

```
X-Hashcash:
1:20:180401:sandipc@cse.iitkgp.ac.in::0000000267674
b591257b87:6078
```

- Version: number of zero bits in the hashed code: date: resource: optional extension: string of random characters: counter

So, here is an example of a hashcash. So, the hashcash which is included in a email header it looks like that you have a hashcash failed after the hashcash failed, you have certain individual field. So, the first field one it talks about the version number. So, we are using hashcash version 1 the 20 is certain number of bits were we can say that it is the number of 0 bits that need to be there in the hash code. So, this 20 says that you need to generate hash function where the first 20 bits of the hash function would be 0.

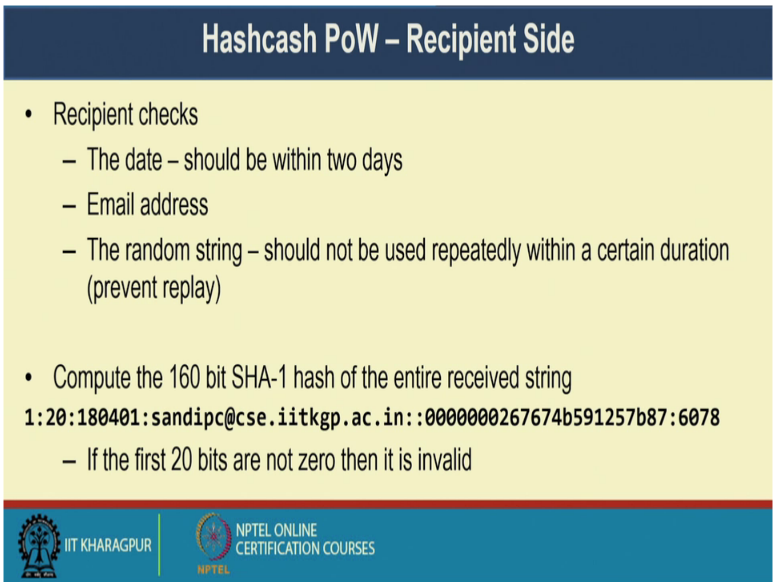
So, that is the challenge which is post to the to the email sender. Now, the email sender has to find out the hash value where the first 20 bits are 0. Then the time stamp value did the here it was generated on first April 2018. So, it is that year month date format 18401, then certain resource here I have put my Email ID.

Then a random number a string of random characters a string of random character is included in the hashcash and what is the importance of the string of random character, that will see later and then a counter value. So, this counter value is working like a nons. So, the email sender for them this all the fields like, the version number of 0 bits, that is timestamp the resource fields the string of random characters all these fields are fixed the only field email sender can change is that counter field.

So, the email sender need to check with different counter value that we by providing different counter value that for which counter value it can find out the expected hash value where the objective is to have a 20 number of 0s at the prefix. So, in the senders side the hashcash is constructed in this way so you first construct the header. So, the construct the header you have individual fields after putting individual fields in the header then the sender initialization counter value to a random number. So, once the sender initialize the counter value to a random number then the sender compute in a traditional hashcash we use 160 bit SHA-1 algorithm.

So, by applying 160 bit SHA-1 algorithm you try to find out a hash value where the first 20 beat of the hash is all 0s. If you can find out a hash value where the first 20 bit of the hash is all 0 than then that particular hash is accepted, otherwise you try with a different counter you change the counter value and check with a different counter value and find out that whether the resultant has produces that required 20 bit at the beginning or not.

(Refer Slide Time: 19:56)



Hashcash PoW – Recipient Side

- Recipient checks
  - The date – should be within two days
  - Email address
  - The random string – should not be used repeatedly within a certain duration (prevent replay)
- Compute the 160 bit SHA-1 hash of the entire received string

`1:20:180401:sandipc@cse.iitkgp.ac.in::0000000267674b591257b87:6078`

  - If the first 20 bits are not zero then it is invalid

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

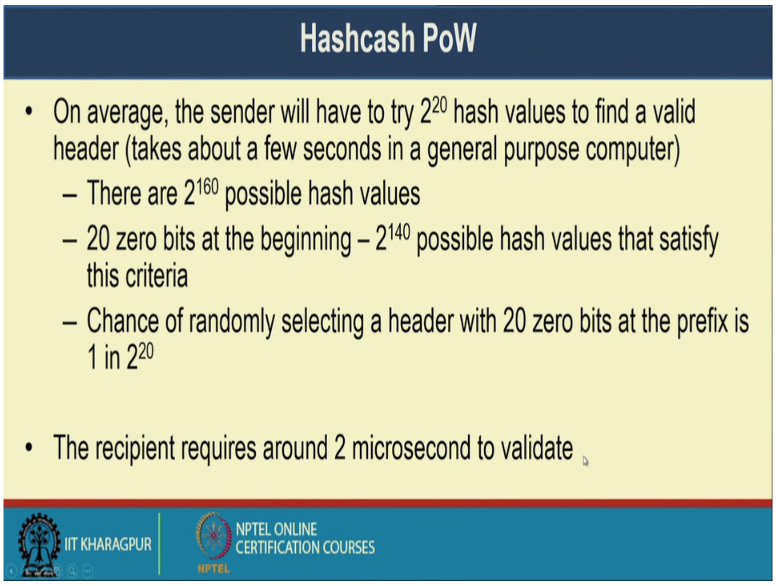
Now, at the recipient the side the recipient checks certain parameters the date the date should be within two days as per the hashcash version 1. So, if someone is changing the date if some email spam or some autocracies making a change in the date version he need to again the compute hashcash, the email address it takes that while the email address in the hashcash it exactly matches with the sender email address.

So; that means, that the sender has generated that particular hashcash value and the random string. So, this random string should not be used repeatedly within a certain duration. So, it prevents the replay attack. So, the replay attack is something like this where the attacker has told the previous hashcash, attacker has cash the previous hashcash and by cashing the previous hashcash with every junk email he or she is sending that previous hashcash.

So, this random string is utilized to prevent those kind of reply after where every time when you are sending an new email you have to change that random string. So, if you change the random string you have to generate a new hashcash value and that way you have to spend certain time to generate the corresponding hashcash.

Now, once you have checked all the fields next time is to validate the hashcash, validating the hashcash is pretty easy used at the interesting find out the 160 bits SHA-1 hash function out of that string and if you can find out that will the first 20 bits are 0, all 0s; that means, that was your earlier objective that you can find out by looking at the string that the first 20 bits should be 0. So, if the first 20 bits are all 0s then you accept that particular email otherwise you discuss that address.

(Refer Slide Time: 21:48)



Hashcash PoW

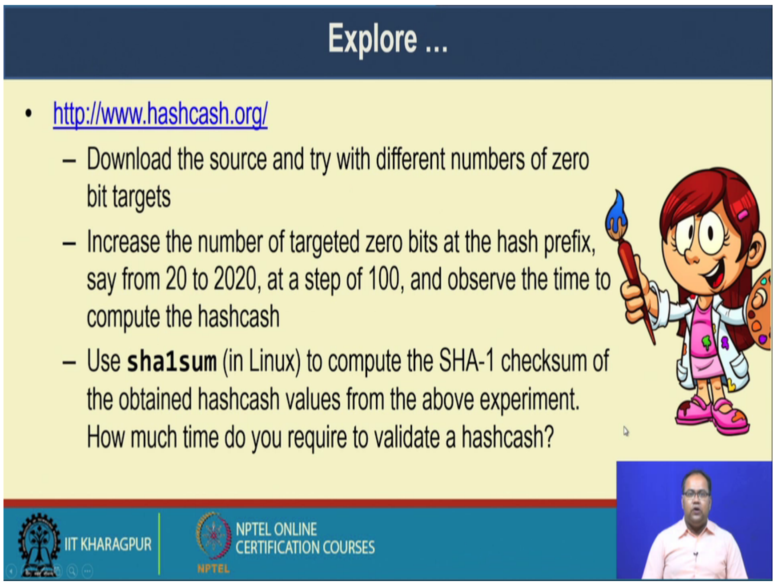
- On average, the sender will have to try 2^{20} hash values to find a valid header (takes about a few seconds in a general purpose computer)
  - There are 2^{160} possible hash values
  - 20 zero bits at the beginning – 2^{140} possible hash values that satisfy this criteria
  - Chance of randomly selecting a header with 20 zero bits at the prefix is 1 in 2^{20}
- The recipient requires around 2 microsecond to validate

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Well some analysis of hashcash proof of work. So, because you are utilizing 160 bits SHA-1 hash function. So, you can have two to the power 160 possible hash values. Now, what we want? That if you want 20 bits at the beginning; that means, you can have 2 to the power 140 possible hash values which satisfy the criteria where the first 20 bits are 0s and remaining 140 bits can be either 0 or 1.

So, the chance of randomly selecting a header with a 20 leading 0 bits at the prefix that probability is 1 out of 2 to the power of 20. So that means, if you can try with 2 to the power 20 different hash function with high probability you can get a hash value. So, that way the expected number of rehash the sender has to do is 2 to the power 20 which takes certain amount of time in today's computer it takes in the order of few seconds to compute the hashcash, but on the other hand validating the hashcash at the receiver side is very fast the receiver requires around two micro second to validate the hashcash, ok.

(Refer Slide Time: 23:05)

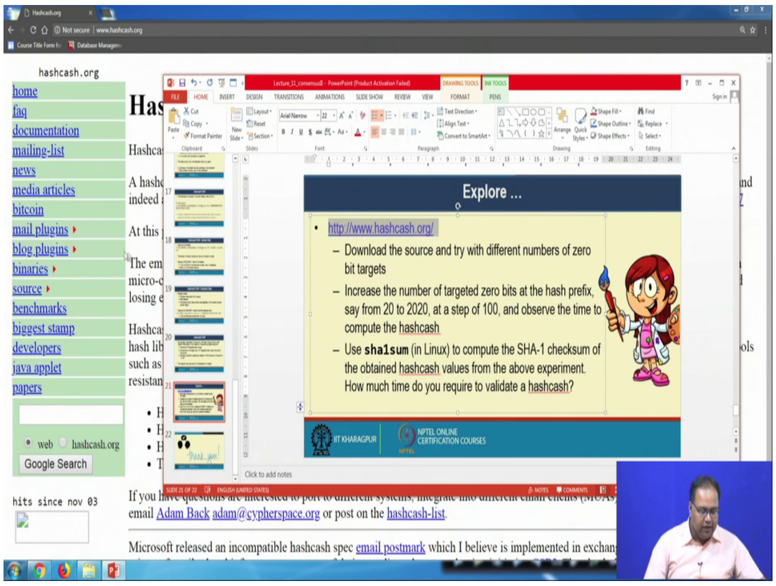


Explore ...

- http://www.hashcash.org/
  - Download the source and try with different numbers of zero bit targets
  - Increase the number of targeted zero bits at the hash prefix, say from 20 to 2020, at a step of 100, and observe the time to compute the hashcash
  - Use **sha1sum** (in Linux) to compute the SHA-1 checksum of the obtained hashcash values from the above experiment. How much time do you require to validate a hashcash?

So, now, there is an interesting exercise for you. So, for these hashcash you can go to the website hashcash dot org.

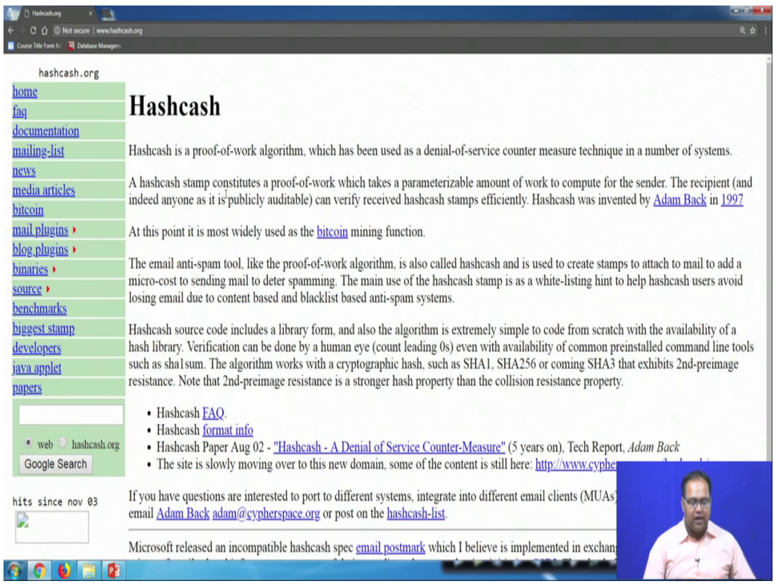
(Refer Slide Time: 23:17)



Explore ...

- http://www.hashcash.org/
  - Download the source and try with different numbers of zero bit targets
  - Increase the number of targeted zero bits at the hash prefix, say from 20 to 2020, at a step of 100, and observe the time to compute the hashcash
  - Use **sha1sum** (in Linux) to compute the SHA-1 checksum of the obtained hashcash values from the above experiment. How much time do you require to validate a hashcash?

(Refer Slide Time: 23:18)



hashcash.org

home
faq
documentation
mailing-list
news
media articles
bitcoin
mail plugins
blog plugins
binaries
source
benchmarks
biggest stamp
developers
java applet
papers

web hashcash.org
Google Search

hits since nov 03

Hashcash

Hashcash is a proof-of-work algorithm, which has been used as a denial-of-service counter measure technique in a number of systems.

A hashcash stamp constitutes a proof-of-work which takes a parameterizable amount of work to compute for the sender. The recipient (and indeed anyone as it is publicly auditable) can verify received hashcash stamps efficiently. Hashcash was invented by Adam Back in 1997

At this point it is most widely used as the bitcoin mining function.

The email anti-spam tool, like the proof-of-work algorithm, is also called hashcash and is used to create stamps to attach to mail to add a micro-cost to sending mail to deter spamming. The main use of the hashcash stamp is as a white-listing hint to help hashcash users avoid losing email due to content based and blacklist based anti-spam systems.

Hashcash source code includes a library form, and also the algorithm is extremely simple to code from scratch with the availability of a hash library. Verification can be done by a human eye (count leading 0s) even with availability of common preinstalled command line tools such as sha1sum. The algorithm works with a cryptographic hash, such as SHA1, SHA256 or coming SHA3 that exhibits 2nd-preimage resistance. Note that 2nd-preimage resistance is a stronger hash property than the collision resistance property.

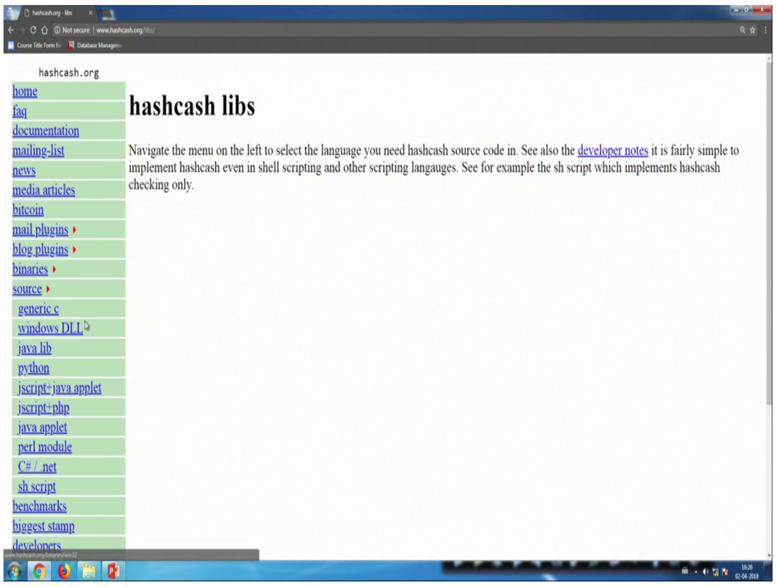
- Hashcash FAQ.
- Hashcash format info
- Hashcash Paper Aug 02 - "Hashcash - A Denial of Service Counter-Measure" (5 years on), Tech Report, *Adam Back*
- The site is slowly moving over to this new domain, some of the content is still here: http://www.cyphe

If you have questions are interested to port to different systems, integrate into different email clients (MUAs email Adam Back adam@cypherspace.org or post on the hashcash-list.

Microsoft released an incompatible hashcash spec email postmark which I believe is implemented in exchan

So, here is the hashcash dot o r g website. In the hashcash dot o r g website you can find out the hashcash source in different languages.

(Refer Slide Time: 23:30)


hashcash.org

home
faq
documentation
mailing-list
news
media articles
bitcoin
mail plugins
blog plugins
binaries
source
generic c
windows DLL
java lib
python
jscript+java applet
jscript+php
java applet
perl module
C# / .net
sh script
benchmarks
biggest stamp
developers

hashcash libs

Navigate the menu on the left to select the language you need hashcash source code in. See also the developer notes it is fairly simple to implement hashcash even in shell scripting and other scripting langauges. See for example the sh script which implements hashcash checking only.

So, under source directory library hashcash sources available in generic c, DLL, java, python, even in perl, C hash, shell script in different languages it is available. So, I will encourage you to download the source the source is very simple and you can just do a direct execution of that. So, executing that particular script I will suggest you to whom follow these exercises. First download with source and try with different number of 0 bit target. So, you can increase the number of targeted 0 bits at the hash prefix say from 20 to some 2020 at a step of 100 and observe the time to compute the hashcash.

So, the output that you will see you will find out that as you increase the number of leading 0 bits in the prefix the time will actually increase exponentially. So, if you increase the challenge, if you increase the complexity of the challenge it will take more time to find out the corresponding hashcash.

So, that way you can determine that at what difficulty you can put the challenge to the email sender if you want that well other email function should have some 100 leading 0 bits. That means, you are actually trying to design a system which we where the sender have to do a lot of work to find out that hashcash, but they will be they but you will be able to guarantee strong security of that system and you will be able to prevent the junk email significantly.

Then under recipient side you use this SHA-1 some script would which is a standard tool available in most Linux computers to compute the SHA-1 checksum of the obtain hashcash value from the above experiment. By doing that SHA-1 sum, SHA-1 sum will return you the 160 bits SHA-1 checksum then you can look into the checksum field and you can find out that well whatever number of bits you have asked for or whatever number of bit 0 bits you have asked for in the prefix weather that many number of 0 bits are there in the prefix or not. And also you can find out that how much time it is taking for the recipient to validate the hash just by applying these SHA-1 some tool on the on the obtained hashcash value.

So, this would be an interesting exercise for you. So, in the next lecture will discuss about bitcoin proof of work which is based on this hashcash based system. So, in today's lecture will stop at this point. So, in the next lecture will see you and will look into the bitcoin proof of work mechanism in further details which is based on the hashcash based system. See you again.

Thank you.