# LSG Work Stream 2022

## LSG / Crypt4GH

**Date: Tuesday, 15 Nov 2022**
**Time: 10:30 - 12:00 pm EST**
**Meeting Chair(s) : Alexander Senf**

Crypt4GH: Enhancing the use of Encrypted Data

Description : Crypt4GH is the GA4GH's encrypted file format standard, designed to enable secure distribution by enabling easy and quick custom encryption, and direct random access to encrypted data. We are now looking to enable new use cases and better integration with other GA4GH standards.

| | **Agenda Item** | **Speaker** | **Time** |
|---|---|---|---|
| 1.0 | Introduction and goals setting | | |
| 2.0 | Crypt4GH: GitHub repo location & regular meeting time review<br>Go to **https://www.slido.com**, type in the event code GA4GH, and then choose the "LSG" room. There is a list of suggested times, or suggest others.Multiple choice is allowed. | Alexander | 5:00 |
| 3.0 | Crypt4GH and htsget. Looking at better ways to enable Crypt4GH data to be delivered via htsget | Alexander | 15:00 |
| 4.0 | Crypt4GH access models and and Cloud standards: access to Crypt4GH data via DRS | Alexander | 40:00 |
| 5.0 | GENXT: Confidential Computing / TEE, data-in-use protection of crypt4gh-encrypted data accessed via DRS | Pavel | 15:00 |
| 6.0 | | | |

**Session: Crypt4GH: Enhancing the use of Encrypted Data**

Tuesday, Nov 15: 10:30 - 12:00 am EST

Zoom Link:
https://us02web.zoom.us/meeting/register/tZIlc-GupjsiE9F7ncQ1r6yAMBIGnfhYYTvO

## Best Practices for Virtual Speakers

- Make sure laptops are plugged into power prior to and during your session.
- Check cameras to ensure they are centered, sit in a well-lit area and ensure your background is what you want for your presentation.
- Use an ethernet cord for the best connectivity; if using Wi-Fi, make sure to test your Wi-Fi connection prior to the conference to ensure it works.
- Earbuds or headphones will prevent audio echoes.
- Please stay muted except when speaking.
- All sessions will be recorded, and the chat boxes will be saved.
- Have water or a beverage close by.

Attendees “Name (Affiliation)”:
Ruslan V,Pavel N,Andrew P,Oliver Hoffman,Rob Davies,James B,Dmitry R,Frederic H,Jeff Liu,Alexander Kanitz,Jamie Delgado,Andres Silva,Geraldine V,Heidi Sofia,Gregori B,Andy Yates,Evan C

Key Takeaways:

- Monthly meeting time poll should be distributed via mailing list to give participant from all time zones a chance to respond.
- Capabilities listed in /service-info would be a good way to add Crypt4GH support without requiring implementation by all projects
- htsget usage patterns may not warrant need for transcoding, but more conversation with htsget is required to address best solution for Crypt4GH support
- There is interest in Crypt4GH support in cloud APIs. Further discussion is needed, also including AAI and Passports

Next action items:

- Plan for April Connect
    - Still have to decide on April Connect meeting
- Plan follow-up meetings

    - Send out poll on meeting time
    - Meeting with htsget: understand usage patterns, discuss the idea of Crypt4GH support as htsget Capability
    - Meeting with Passports/AAI: a comprehensive look at security, and whether Crypt4GH requirements can be met via tokens and/or via information returned by a clearinghouse.
    - Meeting with DRS: SHould happen after the Passports/AAI meeting. Discuss the idea of Crypt4GH as a capability in DRS.
- Any other business arising
    - X
    - x

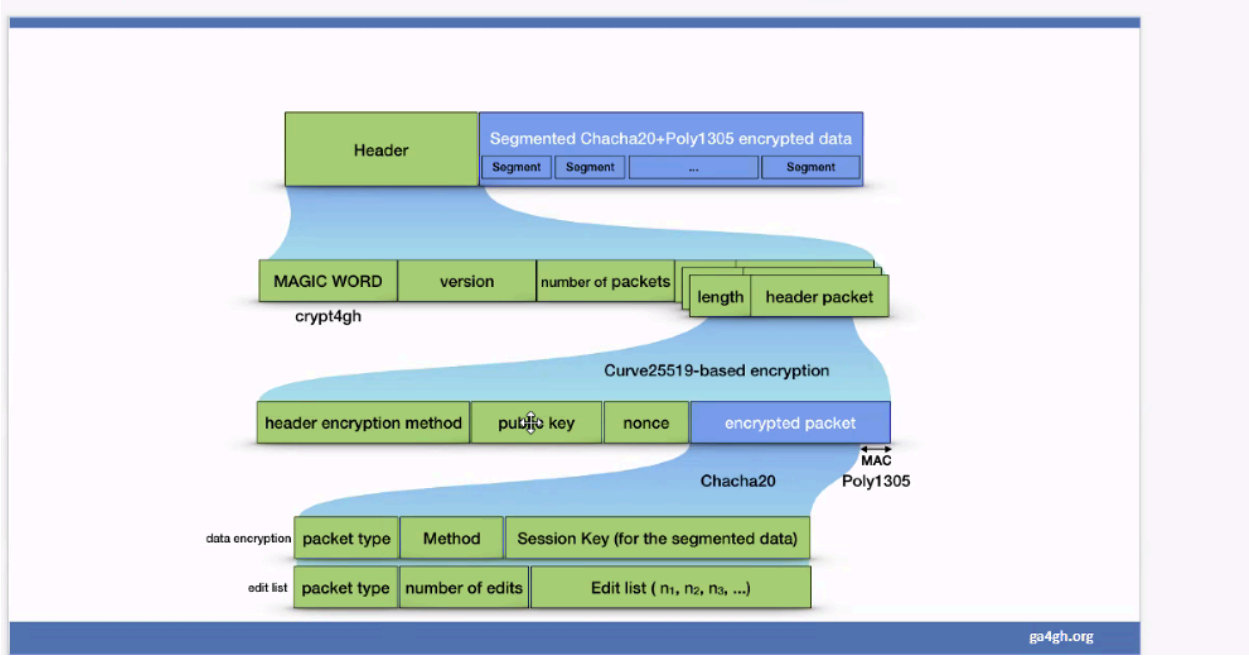
Notes:
Meeting recording :
https://us02web.zoom.us/rec/share/G3ZS68hBAGPIXIUapSxw4wzpzi3rc-gP_Q38YEeP_p5yeuZIiahERMfY2stxHi8W.RC6yPhjwkUWzUMFO?startTime=1668525679000

Meeting transcript - https://otter.ai/u/6zM9zyvL1HhkbLqJV8593i98V_0?f=home

Location where the Crypt4GH code - Need to look into.
AS :Discussion on Htsget
Htsget protocol  - there are few issue we havent resolved yet.
Other one is cloud API in general
new use cases that have developed over the past few years. And one of these use cases is by a company called GENXT

a envelope encryption scheme, where we have a header and is partially unencrypted.
data the payload is encrypted using a symmetric encryption algorithm
The header itself is encrypted with a key that is derived via a public private key scheme.
In this case, it's curve 25519. And what that means is when I encrypt with my private key and your public key, I can generate an encryption key that is the same that you get if you use your private key and my public key.

- The encrypted data itself then contains the session key that you need to access the payload and also the method of encryption used so we it's extendable, but currently we only support ChaCha 22
-




2 type of header - Session key and edit list

- edit list is a direct result of us trying to get Crypt4GH to work with HTS get
- if you look at how the HDS set protocol works. So you have a genomic file, that you're trying to access the region of that file and you query that by genomic coordinates, you want a specific chromosome, you want a specific region of that file, and nothing else. So you send that request to HTSget what HTS get returns to you is a list of URLs that contain the header of the file and the data of the file. You download those independently and simply concatenate them and you have a valid genomic file, this could be a BAM file, a CRAM file, etc.
- The problem we have with crypt4GH in this workflow is we can do the query. So we can say this is the data where you need to send in order to have the data you requested. But what we want to deliver is not the unencrypted data, we want to deliver the encrypted data. So one of the premises of crypt4GH, we deliver an encrypted file to you and we decrypt it as you use it on your local system. However, if you try to get a subset of the file, we are still limited by the 65 or 64k, block sizes of the data packet. So what we have to send to you is all the data packets that overlap the regions that you have queried
- when you simply concatenate the file as you do with HTSget, you have a file that doesn't follow the file format standards, like it's no longer a valid cram file, because you have stuff in the middle that doesn't belong there. And that was the reason that we introduced the edit list, it allows you to skip certain bits of the decrypted data that do not belong to the file.
- Several implementors ignoring edit list feature

- Problem - The other problem is because we send encrypted data out without making any changes to the encrypted data before we send it. We are limited in what we can send. So HTS get supposedly can specify I want a BAM file or I want a cram file, things like that. With crypto GH accounts, you are limited to the file that is given on storage. If you have a crime file, you can only deliver cram files. And so these are the two things we're trying to get a discussion on how we could possibly solve this.

Many standards exist like htsget.
doesn't really make sense to force every single HTSget server to support encrypted data.

AY - the idea of capabilities is definitely in the service info implementation that accompanies ref get, because it's how we describe support for the various different identifier schemes.

In fact, it's service info was originally intended to disseminate this capability information, it's taken on a bit more of a license and to actually send over standardized API information.

AS - we have to implement in order to make it work with HTSget that is to re-encrypt data. So we can stick with the data as it is, and send it out. Because we can't change formats. But if you re encrypted, then we can also get rid of the edit lists, which was the huge win, it would make Crypt4GH easier

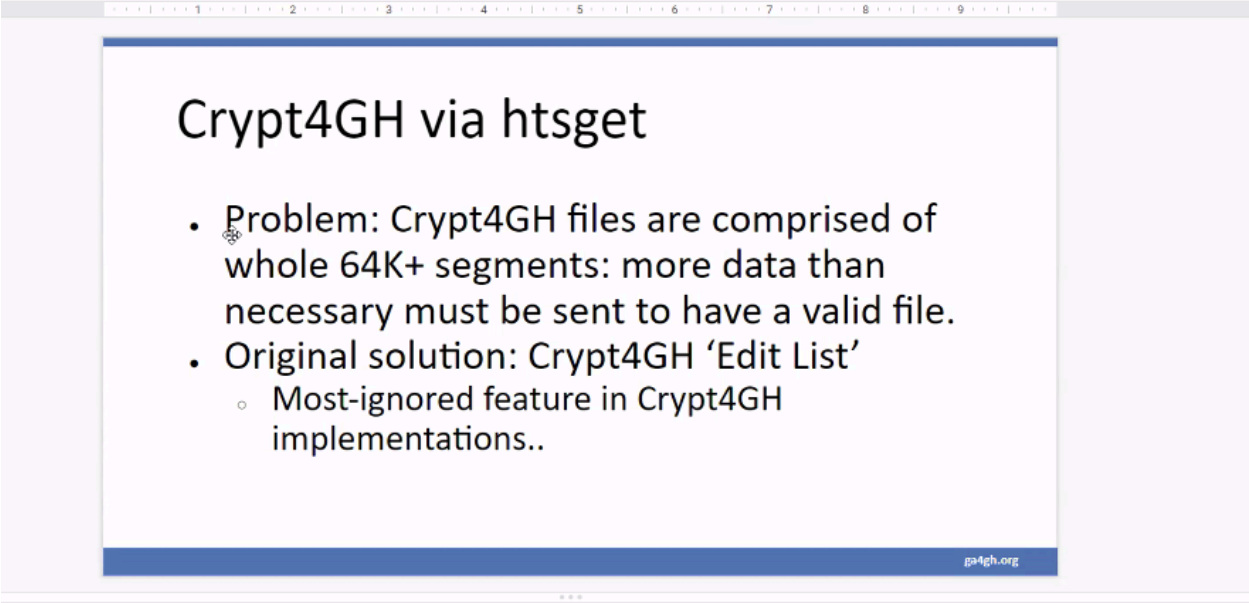
JB - edit list is also used for removing the bits, which are not part of the sort of the overall file format as it were, such as block based if you just do a random block based thing. So would you encrypt the sort of start and end blocks and leave the in between blocks the same or something?
AS - without encryption, we need the edit list, because then we are stuck to the block size, the block boundaries as they are as the file was encrypted. But if you do read encryption, we can actually just sort of pre concatenate the data on the server side and encrypt as one continuous.

JB - Do you need to re-encrypt all of it? Or can you just re encrypt sort of like the boundary blocks and leave the stuff in the middle? I mean, how many different encryption keys do you support for multiple keys for different regions?

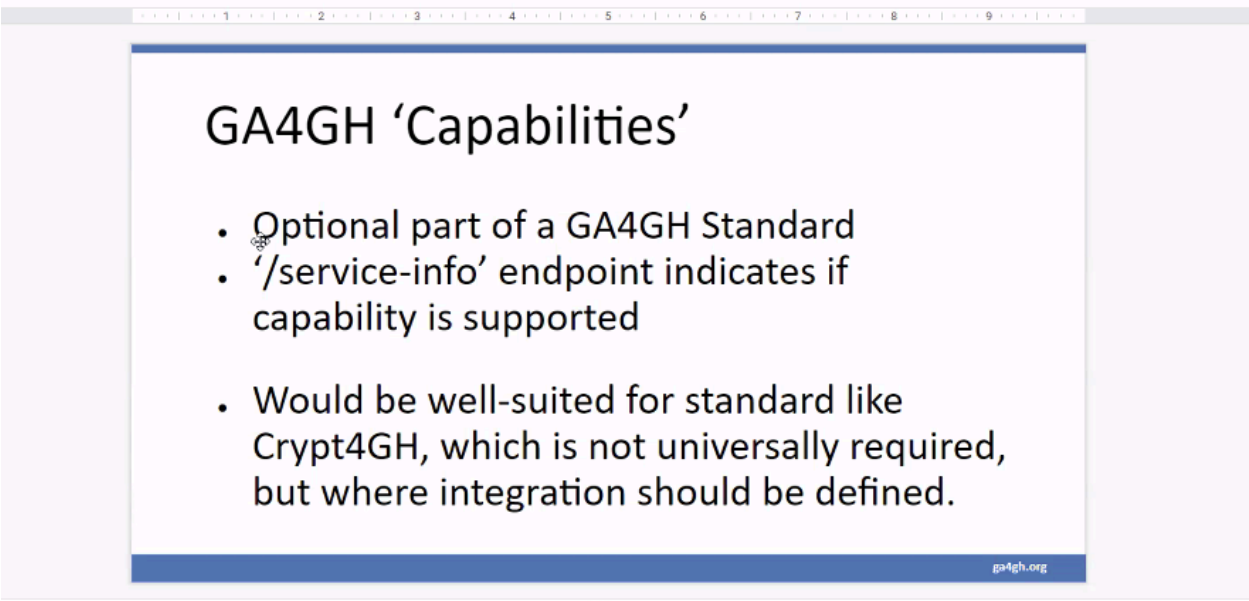
RD - you've got not only your 64k boundaries on the crib for GH, but you've also got the boundaries inside the
RD - maybe you don't want to re-encrypt with a Crypt4GH anyway, you just decrypt it on the server side. And use crypt4GH of where his way of keeping your server data safe from hackers. And then you rely on the TLS encryption to encrypt it,

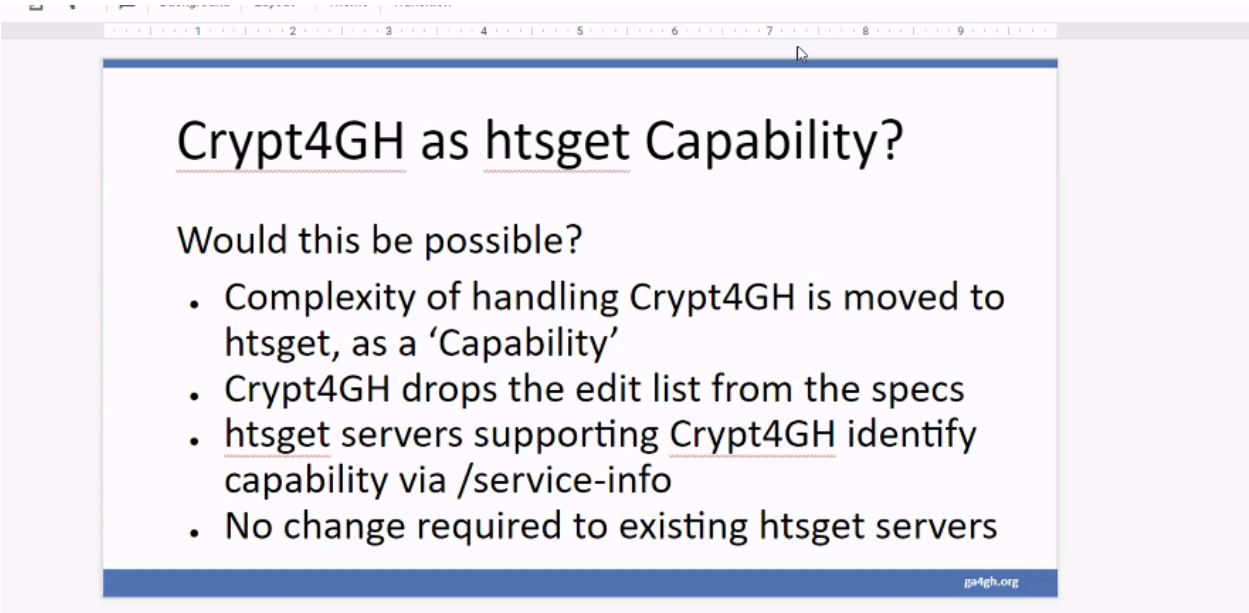
Htsgets return header of the file
Problem - we can do the query, need to deliver encrypted data.

OH - Not suggesting to change it. Not sure we have critical mass between the three people here working on htsget to make a difference.




Limitation - we can deliver only the encrypted file or the cram file as example






JB - Transcoding a region of CRAM to BAM presumably requires re-encrypting, so the htsget server is embedding its own new session key? (As well as having a bunch of work to do, but that's inherent with format change.)
AS: If we re-encrypt we can get rid of the edit list

FH - you also have the decryption on the server side, you want to send the encrypted data and then have the decryption on the client side

RD : having the re- encryption on the server side would be quite a lot more work.
RD - make artificial blocks, which are just the right length to get you back into sync.

FH - Why don't we attach headers to the response. So the implementation of the HTS get server will return to you the file, the blocks that are encrypted. And you are attached to the response headers that says, that's basically the edit list the content of the edit list, but it's not part of the header. It's part of the HTTP response, then it's up to the implementation of the server and then via the service info, it will tell you whether it supports that or not.

RD - Hts get to Drop off too much data
FH - So the content of the edit list that was inside the header, you just take it and you put it outside the header and since it's transferred over TLS you're not exposing anything anyway. And the data is still encrypted at rest.

DR : Fred are you talking about using HTTP Header or multy-mime?
FH : we are talking about HTTP headers.
FH: we can attach the header at the end once done streaming.
RD: info on the response header?
FH: preferred to have decryption at the client side.
OH: how much tied to transcoding? How many HTS get users currently use transcoding?

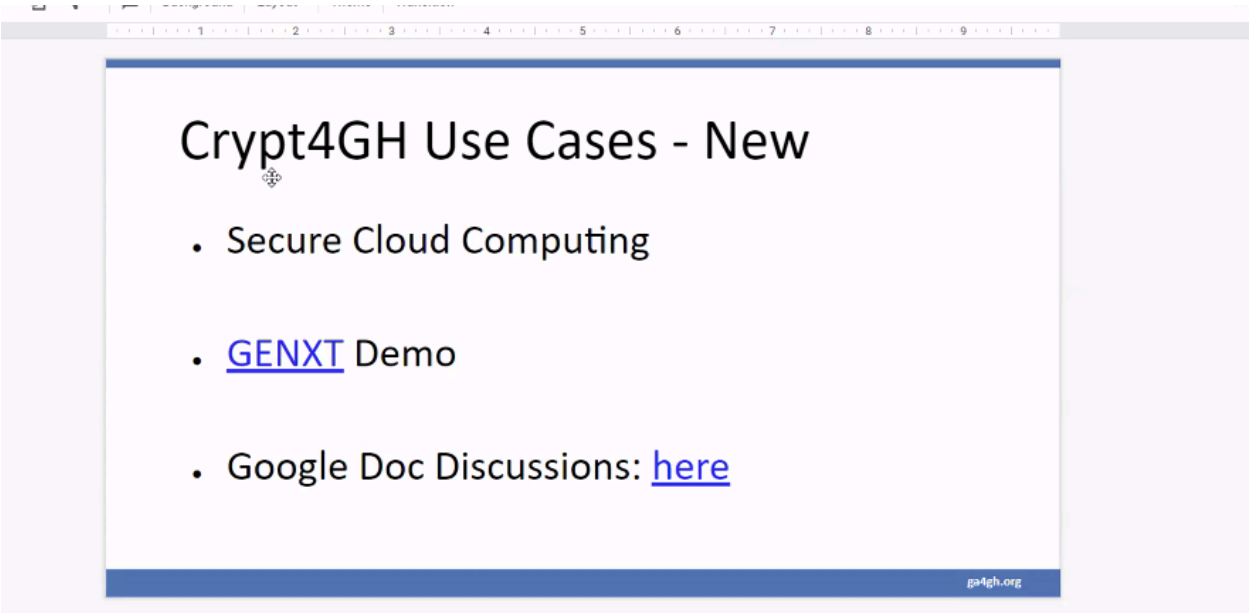
don't know if anyone who was looking into that as a capability or as a feature that will be a requirement. If there's a CRAM that will just accept the crown. There was no transcoding plan generally heard externally ever since so much that's been used.

AS: not sure how much has been used.
RB: Jason Using range request. It won't do the transcoding.

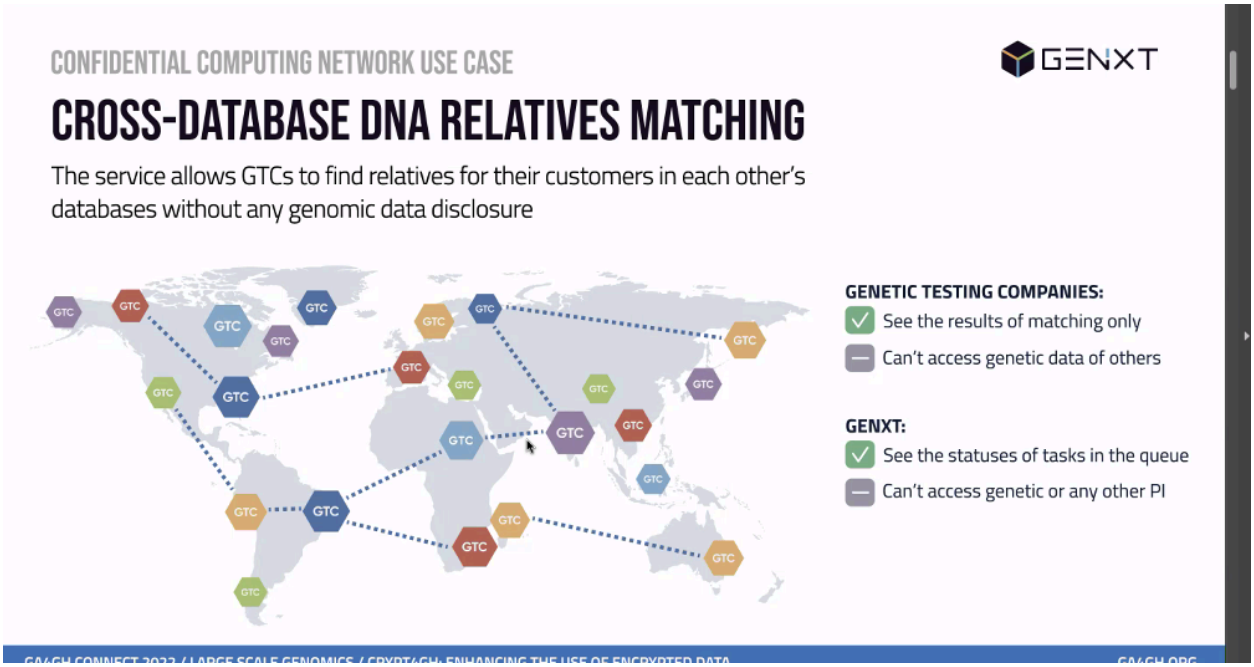
As: If we are not using transcoding, we continue to use edit listing as if we're not losing anything, except that it's not a popular feature as implementers of GA4GH. So the question might be, how much of that should be outsourced to HTS get server, if any, at all?
JB - Tricky for EBI to answer this maybe as 90% of the data stored is in CRAM.  Needs multiple htsget implementers to track

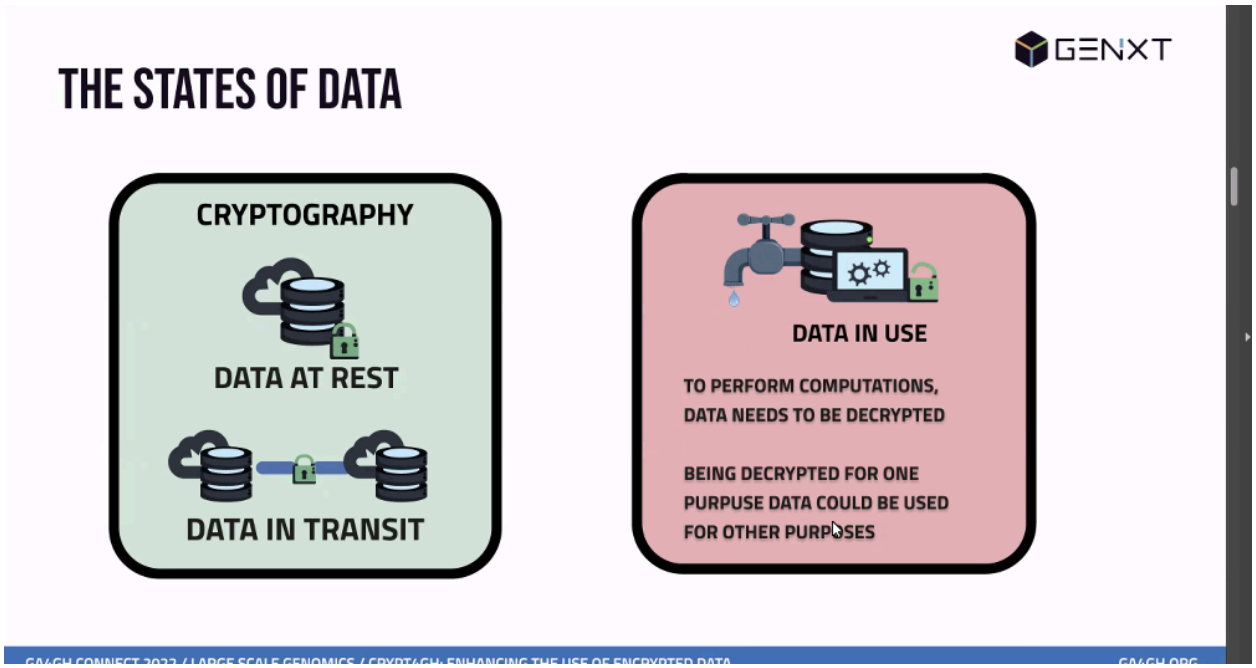
Crypt4 GH Use case:
AS: deliver the encrypted to the user and crypt4gh making it easier. This is how we started out.

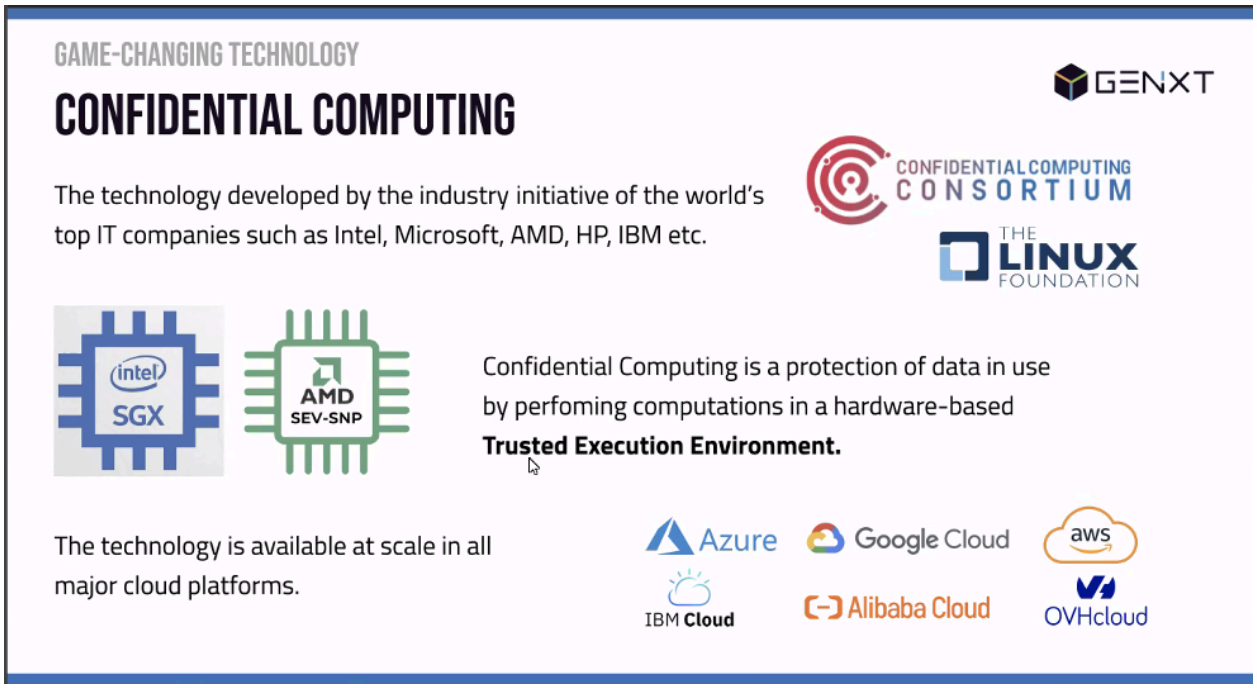
AS - In the past, that meant re encrypting the entire file, the entire amount of data that you requested would be re encrypted using a key that the user either supplied to us or that we would like to give out to the user
As - use cases have in common is that there is no longer this reliance on a particular key that is tied to a particular user. So that's, that's use cases where we may have key paths where nobody knows the private key except a certain server within a Trusted Execution Environment




GENXT demo (slides):





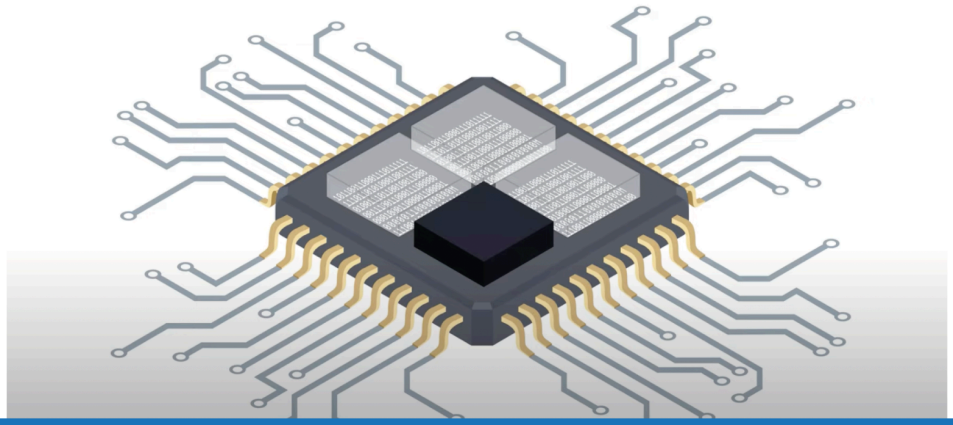

Process memory is no longer accessible to admin

GAME-CHANGING TECHNOLOGY

GENXT

# TEE: MEMORY ISOLATION

Process memory is no longer visible for the operating system and administrator.



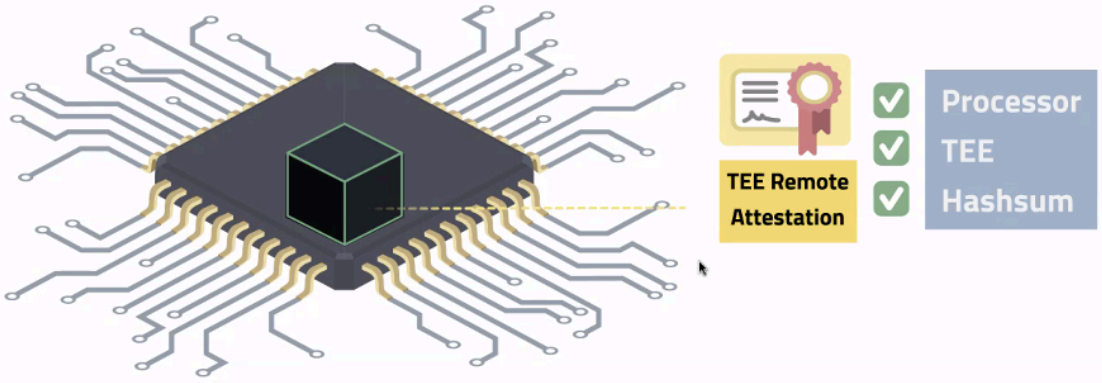


GAME-CHANGING TECHNOLOGY

GENXT

# TEE: REMOTE ATTESTATION

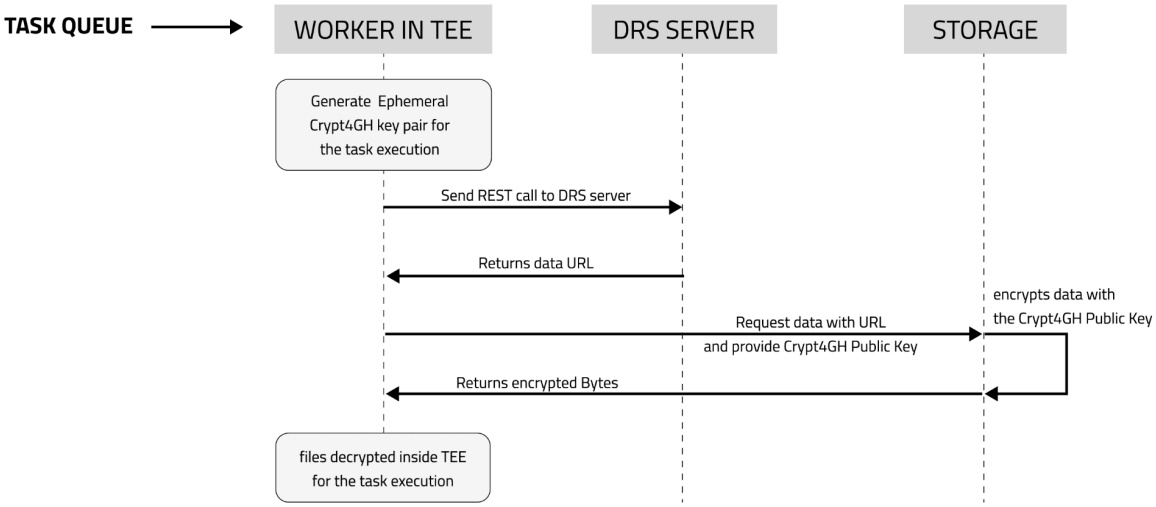
Verifier can check the remote TEE binary hashsum and obtain a unique secret within the procedure.



PROTOCOL OVERVIEW

## CRYPT4GH DATA ACCESSED IN TEE VIA DRS

GENXT

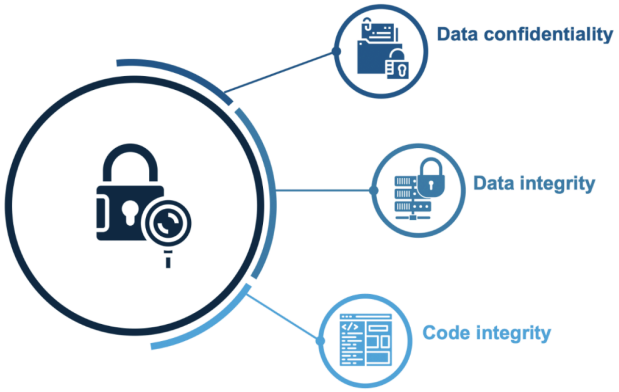





GAME-CHANGING TECHNOLOGY

GENXT

## CONFIDENTIAL COMPUTING AND CRYPT4GH ENABLE PRIVACY-BY-DESIGN DISTRIBUTED SYSTEMS






As : Integration with the cloud
From the DRS point of view how could we integrate the capability

HS : Is the Work Order Token proposed for GA4GH Passports 2.0 relevant to Crypt4GH for fine-grained access control?
AS : Combine TEE environment with Work order token?
HS : Need to have discussion with passport, cloud and htsget group