CGN Data Flow with Security Explained (v2.01)

Introduction

The audience of this document is anyone in the enterprise who wishes to investigate the possibility of using the CGN ecosystem for processing high volumes of very large, complex computational workloads as an alternative to dependence on costly cloud compute providers, and who wishes to understand how the CGN keeps data secure for meeting enterprise security requirements.

This document illustrates the workflow enterprise dApps will use to communicate with its remote dServices to securely process large data sets using the CGN. We explain how enterprise data is kept secure at each stage of the way as data courses through the system, is processed, and makes its way back to the enterprise dApp.

Because this information pertains to the secure enterprise use case, only Tier-1 and Tier-2 miners, i.e. data center spare capacity, are considered. The CGN will never let consumer-grade miners process sensitive enterprise data, so all discussion of Tier-3 and Tier-4 miners is omitted here.

Miner Tier Properties			
Tier	Location	Trusted	Use Case
Tier-1	Corporate (any)	Yes	Private computing (including clusters)
Tier-2	Corporate, data center or bulk hardware	Yes	High performance, high capacity computing (including clusters)
Tier-3	Consumer (mostly mobile)	Yes	One-off enterprise bulk work where security, privacy and correctness are paramount
Tier-4	Consumer (all devices)	-	One-off consumer space bulk work

The Secure Enterprise Use Case (Large Data Sets)

1. An enterprise dApp is to be hosted on enterprise servers, using the CGN at scale to perform many computationally expensive workloads in parallel, over time.
2. A corresponding enterprise dService is to be hosted within the CGN by virtue of being replicated onto many miners.
3. Work operations will separately reference large Work Input Data to be fetched by the dService as a large binary or text file. Work Output Data will also be returned as a large binary or text data file. For scalability, these data files should not pass through the CGN. Instead, references to these data files will be passed to the miner through the CGN. *Note: This is distinct from the Small Data Set use case, where work inputs and/or outputs can be transmitted directly, together with the request/response pair.*
4. Security will be of utmost importance to the enterprise. It's critical that data be protected from snooping or theft at every stage of work execution and data transfer. To eliminate risks arising from the possibility of byzantine miners, only Tier-1 and Tier-2 (data center) miners will be selected by the CGN to perform this work. Additionally, the enterprise dService will be enabled only on certain, specific miner instances. These constraints are fully specified in the Enterprise dApp and Miner Operator account records and will be respected by the CGN.

Types of Data

1. DApp Work Command Request to be sent to the dService
   - This is a work request command sent from the enterprise dApp to the CGN, for assignment to an available miner. The request is sent as an HTTP POST request to the CGN, which is then relayed securely to the selected miner. Such requests will contain a command for the dService to execute

and values for all required input parameters. Request parameter names and meanings are specified by the chosen enterprise dApp/dService command communication protocol.
    - The input POST parameters may contain a URL reference to a work data input file hosted from the enterprise which the miner can read.
    - Any authentication information needed by the miner to fetch the work input data file may be passed in additional POST parameters or may have been established *a priori* with the miner. The specific approach used is determined by the design of the enterprise application.
2. DService Work Input Data
    - This is input data request of the work unit execution.
    - The format and structure of this data are specified by the enterprise dApp/dService pair.
3. DService Work Output Data
    - This is output data response of the work unit execution.
    - The format and structure of this data are specified by the enterprise dApp/dService pair.
4. DService Work Command Response back to the dApp
    - This is the HTTP response corresponding to the request in step 1.
    - The response will include whatever information the dApp expects to receive from the dService, stating that the work was received and will be started, and may include additional information about the work output such as its generated filename. The format and structure of this response data are specified by the enterprise dApp/dService command communication protocol.

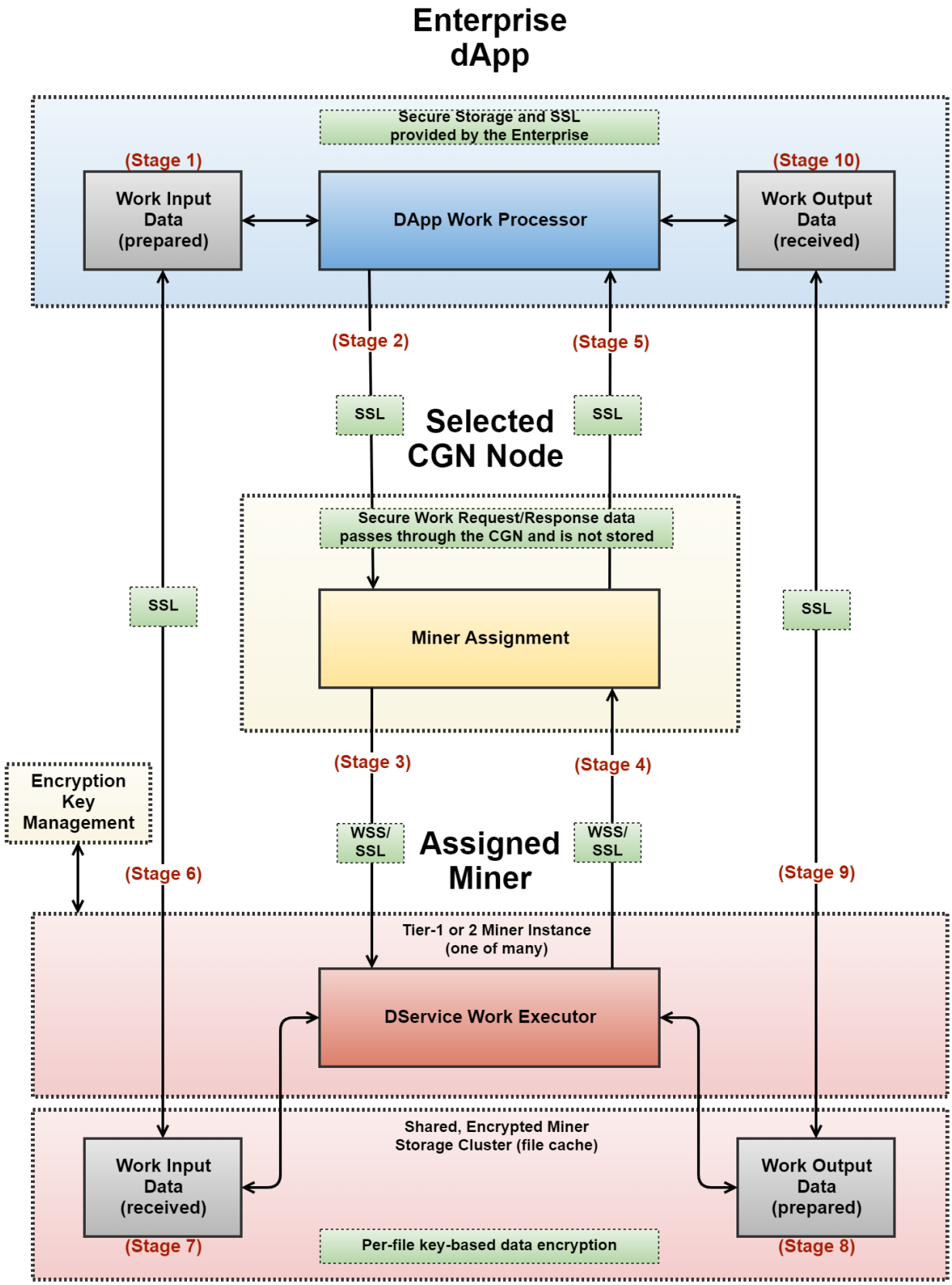
Data Processing Workflow Steps

1. Enterprise dApp prepares Work Input Data and places it in an externally accessible location where miners can download it.
2. Enterprise dApp issues Work Command Request to the CGN.
3. CGN selects a miner and forwards the Work Command Request to the miner.
4. Miner receives the Work Command Request from the CGN and prepares a suitable Work Command Response. The work is initiated.
5. Miner sends the Work Command Response to the CGN, which forwards it to the Enterprise dApp.
6. Miner checks to see if all needed Work Input Data file(s) already exists in the local cache.
7. If the Work Input Data file(s) aren't found in the cache, the miner downloads missing Work Input Data file(s) from the Enterprise server and stores them in the local cache.
8. Once all Work Input Data is fully received at the miner, processing is begun, and the Work Output Data file is generated.
9. When complete, Work Output Data is uploaded to the Enterprise dApp as per the Work Command Request.
10. Once Work Output Data is received at the Enterprise dApp, the dApp proceeds to make use of the results.

Data Movement Stages

1. Prepared Work Input Data- at rest, on enterprise's own servers
2. Work Command Request- in transit, from enterprise dApp to CGN
3. Work Command Request- in transit, from CGN to miner
4. Work Command Response- in transit, from miner to CGN
5. Work Command Response- in transit, from CGN to enterprise dApp
6. Work Input Data- in transit from the enterprise dApp to the miner
7. Received Work Input Data- at rest on the miner's filesystem
8. Prepared Work Output Data- at rest on the miner's filesystem
9. Work Output Data- in transit from miner to enterprise dApp
10. Received Work Output Data- at rest on enterprise's own servers

Enterprise Data Flow Diagram Showing Security Measures Taken



Security Explanation for each Data Movement Stage

Stages 1, 10: Work input and output data at rest, on enterprise's own servers

Threats:

- As determined by the enterprise.

Mitigation:

- To be implemented by the enterprise.

Stages 2, 5: Command request round trip transit between enterprise dApp and CGN

Threats:

- Data theft by Man In The Middle (MITM) attack.

Mitigation:

- Data is encrypted using SSL associated with the CGN domain name certificate.

Stages 3, 4: Command request round trip transit between CGN and miner

Threats:

- Data theft by Man In The Middle (MITM) attack.
- Fake or compromised miner.

Mitigation:

- Data is transmitted over private WebSocket connection initiated by miner into the CGN.
- Data is encrypted using WSS/SSL associated with the CGN domain name certificate.
- Miner is hosted securely in a Tier-1 or Tier-2 datacenter, where it cannot be compromised.

Stages 6, 9: Work input data round trip transit between enterprise dApp and miner

Threats:

- Data theft by Man In The Middle (MITM) attack.

Mitigation:

- Data is encrypted using SSL associated with the enterprise domain name certificate.

Stages 7, 8: Work input and output data at rest, on the miner's filesystem

Threats:

- Inspection of data by users with access to the miner host machine.
- Theft of data by successful intrusions into the miner host machine.

Mitigation:

- Data is symmetrically encrypted using a data-specific (per-file) encryption key, as assigned by the Encryption Key Management component. The key is sent to a miner only when needed and is never saved on a miner. Therefore, even if data cached on miner is stolen, attackers cannot read it without the key.
- All work files stored by miners are subject to a data retention policy, which is specified by enterprise account settings. The miner client will respect the specified TTL by terminating retention of expired Work Data files.

Work Data Retention

1. Work Input Data and Work Output Data having potential for reuse by future Work Command Requests will be persisted on the miner or miner cluster file system in a shared cache area where the miner or other miners in the cluster can access the data for reuse.
2. Work Input Data and Work Output Data having no potential for reuse will have a TTL of zero, meaning it will not be retained.
3. All cached data will be kept only until its TTL (time to live), which is established in advance by enterprise account settings or on the specific Work Request. Expired data on the miner instance will automatically be securely deleted. Furthermore, its encryption key is also deleted for additional protection.

Account Credentials Management

Account Records

Our Central Authentication Server provides authentication services to the CGN, which then provides a subset of those authentication services to the miner. All ecosystem components are created with an account record, protected by credentials:

Ecosystem Components having an Account Record:

1. For each CGN instance
2. For each Miner Operator
3. For each Enterprise operating a dApp
4. For each authorized child user of the Enterprise account
5. For each dApp/dService Developer

Users of each account type can access the ecosystem in two distinct ways:

1. By API key, to make use of the ecosystem
2. By account password, to administer an account

Ecosystem Component	Role of API key
CGN instance	Required for CGN to obtain private information from Central Server
Miner instance	Required for miner to join the CGN and participate by doing work
Enterprise account	Required for enterprise to be able to submit work to the CGN
Enterprise authorized user	Used for tracking enterprise user work actions performed by the dApp
dApp/dService developer	(TBD)

Note that because each miner authenticates with the CGN, which then forwards the authentication request to central, two separate authentications are performed simultaneously:

1. The CGN account ID + CGN API key lets Central authenticate the CGN.
2. The Miner account ID + Miner API key lets Central authenticate the Miner on behalf of the CGN.

Thus, the Central Server authentication endpoint will be secure from brute force attacks.

Password Hashing and Storage

The Central Authentication Server never persists plaintext passwords or API keys. When an account is created, its password or API key are only known only by the component whose account was just created. It is then up to that component to protect its credentials, as they cannot later be recovered from the Central Credentials Database.

The Central Authentication Server persists a one-way hash of credentials in the Central Credentials Database. This is of critical importance in modern security architectures. *Why is it so critical to persist only hashed credentials?* There have been numerous stories in the news about data breaches at companies where hackers made off with loads of user information, often including plaintext passwords. It is a terrible practice to store plaintext passwords anywhere, ever. We use industry best practices in our system that guarantee passwords are never stored, are never logged, and are impossible to recover. Even we can't recover a password from our own data. At best we can only reset a password.

We use the latest and most secure hash algorithm, Argon2, a key derivation function that was selected as the winner of the Password Hashing Competition in July 2015. This hash method by design incurs a maximum of time, memory and parallelism costs, making individual hashes computationally very expensive to generate. To compute a single password hash requires use of most available CPU and RAM resources on the authentication server for roughly 0.1 – 0.5 seconds.

We use a variant called Argon2id, which combines Argon2i (the safest against side-channel attacks), with Argon2d (which provides the highest resistance against GPU cracking attacks).

Because of the extremely high cost to generate even a single hash, brute force attacks to try to recover a password from its Argon2 hash are computationally not feasible.

Additional Measures Taken

We take additional measures to ensure our hashed credentials store will never be breached. We store hashed credentials in a database that is locally network reachable only by the Central Authentication Server and is otherwise isolated from the internet. To maliciously access the database would require obtaining physical access to the machine. *This database has a minimal attack surface.*

The Central Authentication Server itself only opens two ports, 6661 and 6443 for HTTP POST requests to service ecosystem authentication needs, which minimizes its own attack surface. There are no other open ports that could be used to break into the Central Authentication Server. As with the database server, you would need to obtain physical access to the Central Authentication Server machine itself to attempt any malicious action.