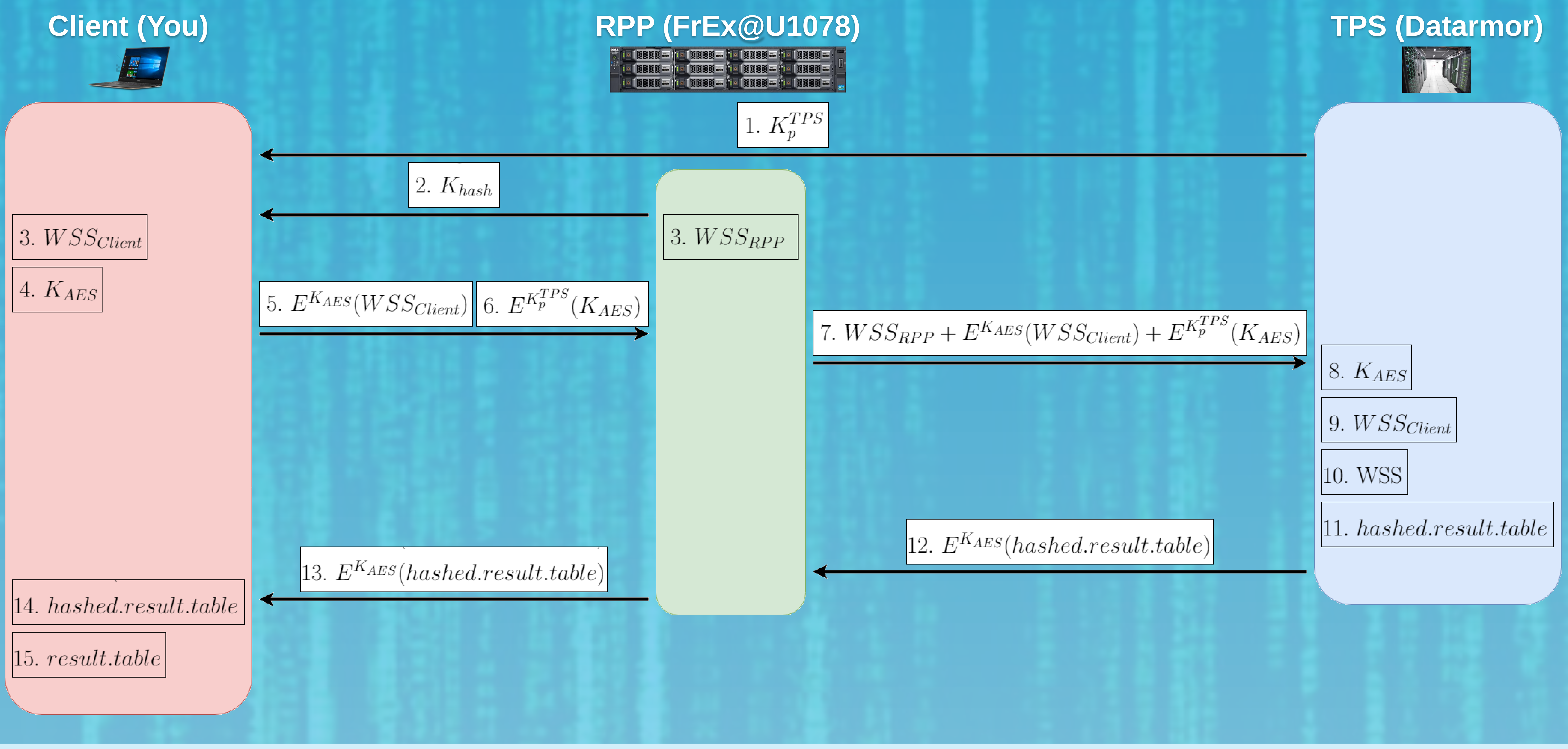
PrivAS: a tool to perform Privacy-Preserving Association Studies

Thomas E. Ludwig[1,2], Reda Bellafqira[3], David Niyitegeka[3], Daniel Salas[4],
Isabelle Perseil[4], Gouenou Coatrieux[3] and Emmanuelle Génin[1]

PrivAS is a tool to perform Genome-Wide Association studies (GWAS) using the Weighted-Sum Statistic (WSS) algorithm in a Privacy-Preserving environment. The underlying scenario takes into account three interacting parties: (1) a Client, e.g. a genomic research unit, wanting to measure the association between an observed phenotype and regions of the genome; (2) a Reference Panel Provider (RPP) possessing genetic data for a Reference Panel, e.g. a priori healthy individuals of a carefully selected ancestry and (3) a Third-Party Server (TPS) with large computational capacities. Our tool and its underlying implementation preserve both state-of-the-art performances and Privacy for all parties. Indeed, through a series of hashing and encryption mechanisms, we can assure that no genetic data from neither the Client nor the RPP are visible by the other parties involved. Furthermore, only the Client is able to view a decrypted version of the WSS results.

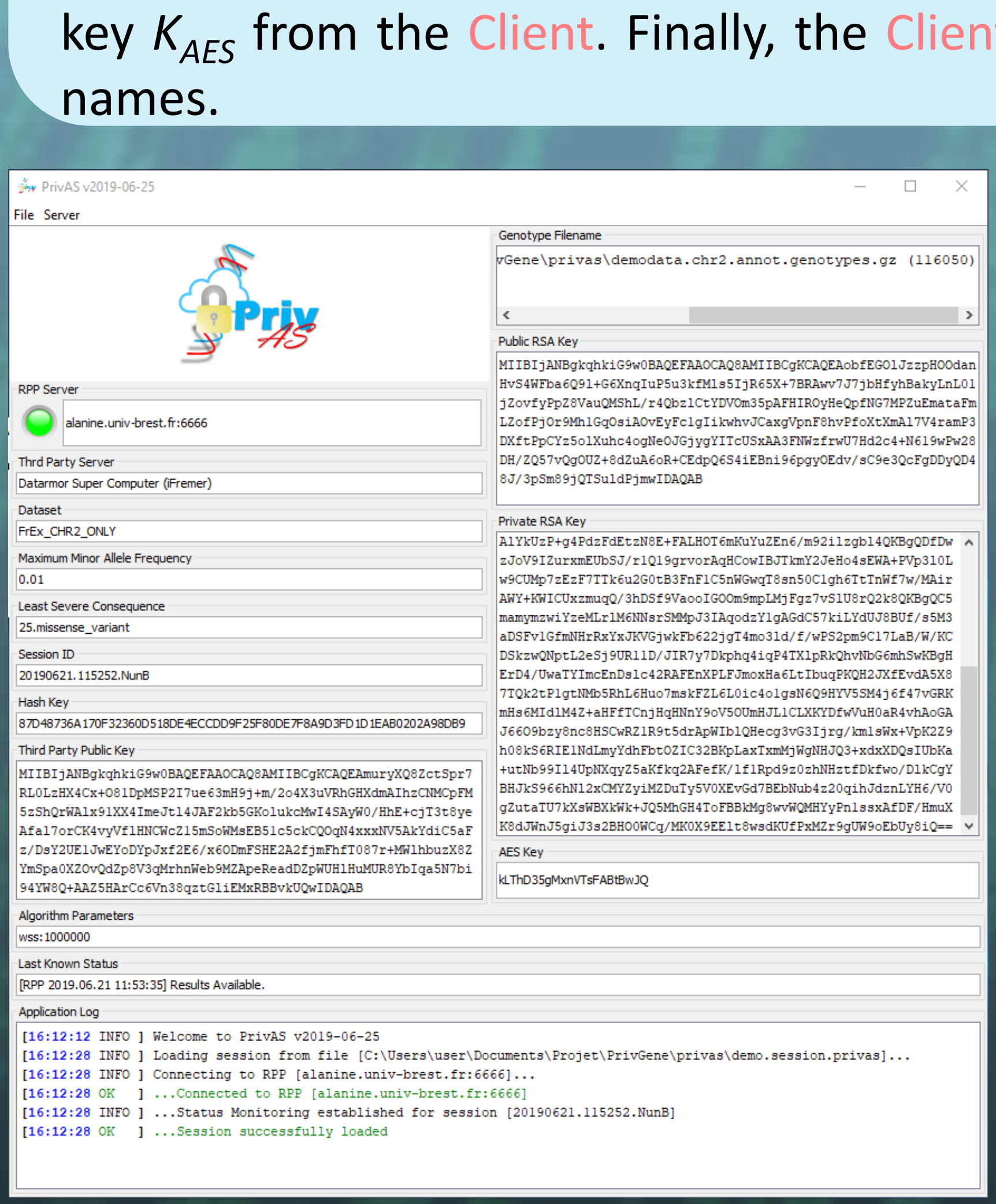

Client (You)
RPP (FrEx@U1078)
TPS (Datarmor)
1. K_p^{TPS}
2. K_{hash}
3. WSS_{Client}
4. K_{AES}
3. WSS_{RPP}
5. $E^{K_{AES}}(WSS_{Client})$
6. $E^{K_p^{TPS}}(K_{AES})$
7. $WSS_{RPP} + E^{K_{AES}}(WSS_{Client}) + E^{K_p^{TPS}}(K_{AES})$
8. K_{AES}
9. WSS_{Client}
10. WSS
11. $hashed.result.table$
12. $E^{K_{AES}}(hashed.result.table)$
13. $E^{K_{AES}}(hashed.result.table)$
14. $hashed.result.table$
15. $result.table$

In our implementation of the secure WSS, three parties are involved:

1. the Client that possesses data for individuals presenting the studied phenotype
2. the Reference Panel Repository (RPP) that has data for unaffected individuals
3. the Third-Party Server (TPS) that will do the actual computation.

In order to allow these parties to work together without compromising the privacy of the data, encryption and hashing mechanisms will be implemented. The TPS will execute the WSS algorithm data where the variant its gene's name have been hashed, using the *SHA256* algorithm initialized with a key K_{hash} shared by the Client and the RPP but unknown to the TPS. As the Client doesn't have direct access to the TPS, its data will transit through the RPP server. Since the RPP knows K_{hash}, it is able to intercept the Client's data. So, these data are encrypted using the *AES* algorithm with a key K_{AES} generated by the Client. As the TPS needs to be able to decipher the Client's data, the Client sends K_{AES} to the TPS via the RPP, protecting the key from RPP by using an *RSA* encryption. The Client uses the public *RSA* key from the TPS K_p^{TPS} and encrypts K_{AES} with it. Later the TPS uses its secret *RSA* key K_s^{TPS} to decrypt the message. Once all computations are done, the TPS sends the results (that contain hashed gene names and their estimated p_{value}) to the Client via the RPP. The results are encrypted using the *AES* key K_{AES} from the Client. Finally, the Client decrypts the results and unhashes the gene names.

1. Client gets RSA K_p^{TPS} from TPS
2. Client gets the session's unique SHA256 hash key K_{hash} from RPP
3. Client and RPP use K_{hash} to hash variants and gene names, producing WSS_{Client} and WSS_{RPP}, Client builds hash dictionary
4. Client generates a unique AES key K_{AES}
5. Client uses K_{AES} to encrypt WSS_{Client} and sends $E^{K_{AES}}(WSS_{Client})$ to RPP
6. Client uses K_p^{TPS} to encrypt K_{AES} and sends $E^{K_p^{TPS}}(K_{AES})$ to RPP
7. RPP sends WSS_{RPP}, $E^{K_{AES}}(WSS_{Client})$ and $E^{K_p^{TPS}}(K_{AES})$ to TPS
8. TPS uses RSA K_s^{TPS} to retrieve K_{AES}
9. TPS uses K_{AES} to retrieve WSS_{Client}
10. TPS performs WSS association tests for each $hash^{K_{hash}}(gene)$
11. TPS produces a $hashed.result.table$, listing each $hash^{K_{hash}}(gene)$ to its WSS p_{value}
12. TPS uses K_{AES} to encrypt $hashed.result.table$ and sends $E^{K_{AES}}(hashed.result.table)$ to RPP
13. RPP sends $E^{K_{AES}}(hashed.result.table)$ to Client
14. Client uses K_{AES} to retrieve $hashed.result.table$
15. Client uses hash dictionary on each $hash^{K_{hash}}(gene)$ to get $result.table$



Genotype Filename
```
vGene\privas\demodata.chr2.annot.genotypes.gz (116050)
```

Public RSA Key
```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAobfEGO1JzzpHOOdan
HvS4WFba6Q91+G6XnqIuP5u3kfMls5IjR65X+7BRAwv7J7jbHfyhBakyLnL01
jZovfyPpZ8VauQMShL/r4Qbz1CtYDVOm35pAFHIROyHeQpfNG7MPZuEmataFm
LZofPjOr9Mh1GqOsiAOvEyFclgIikwhvJCaxgVpnF8hvPfoXtXmAl7V4ramP3
DXftPpCYz5olXuhc4ogNeOJGjygYITcUSxAA3FNWzfrwU7Hd2c4+N619wPw28
DH/ZQ57vQgOUZ+8dZuA6oR+CEdpQ6S4iEBni96pgyOEdv/sC9e3QcFgDDyQD4
8J/3pSm89jQTSuldPjmwIDAQAB
```

Private RSA Key
```
AlYkUzP+g4PdzFdEtzN8E+FALHOT6mKuYuZEn6/m92ilzgbl4QKBgQDfDw
zJoV9IZurxmEUbSJ/r1Q19grvorAqHCowIBJTkmY2JeHo4sEWA+PVp310L
w9CUMp7zEzF7TTk6u2G0tB3FnF1C5nWGwqT8sn50Clgh6TtTnWf7w/MAir
AWY+KWICUxzmuqQ/3hDSf9VaooIGOOm9mpLMjFgz7vS1U8rQ2k8QKBgQC5
mamymzwiYzeMLr1M6NNsrSMMpJ3IAqodzY1gAGdC57kiLYdUJ8BUf/s5M3
aDSFvlGfmNHrRxYxJKVGjwkFb622jgT4mo31d/f/wPS2pm9C17LaB/W/KC
DSkzwQNptL2eSj9UR11D/JIR7y7Dkphq4iqP4TXlpRkQhvNbG6mhSwKBgH
ErD4/UwaTYImcEnDslc42RAFEnXPLFJmoxHa6LtIbuqPKQH2JXfEvdA5X8
7TQk2tPlgtNMb5RhL6Huo7mskFZL6L0ic4olgsN6Q9HYV5SM4j6f47vGRK
mHs6MId1M4Z+aHFfTCnjHqHNnY9oV5OUmHJL1CLXKYDfwVuH0aR4vhAoGA
J6609bzy8nc8HSCwRZ1R9t5drApWIblQHecg3vG3Ijrg/kmlsWx+VpK2Z9
h08kS6RIElNdLmyYdhFbtOZIC32BKpLaxTxmMjWgNHJQ3+xdxXDQsIUbKa
+utNb99Il4UpNXqyZ5aKfkq2AFefK/1flRpd9z0zhNHztfDkfwo/DlkCgY
BHJkS966hNl2xCMYZyiMZDuTy5V0XEvGd7BEbNub4z20qihJdznLYH6/V0
gZutaTU7kXsWBXkWk+JQ5MhGH4ToFBBkMg8wvWQMHYyPnlssxAfDF/HmuX
K8dJWnJ5giJ3s2BHO0WCq/MK0X9EElt8wsdKUfPxMZr9gUW9oEbUy8iQ==
```

AES Key
kLThD35gMxnVTsFABtBwJQ

RPP Server
alanine.univ-brest.fr:6666

Thrd Party Server
Datarmor Super Computer (iFremer)

Dataset
FrEx_CHR2_ONLY

Maximum Minor Allele Frequency
0.01

Least Severe Consequence
25.missense_variant

Session ID
20190621.115252.NunB

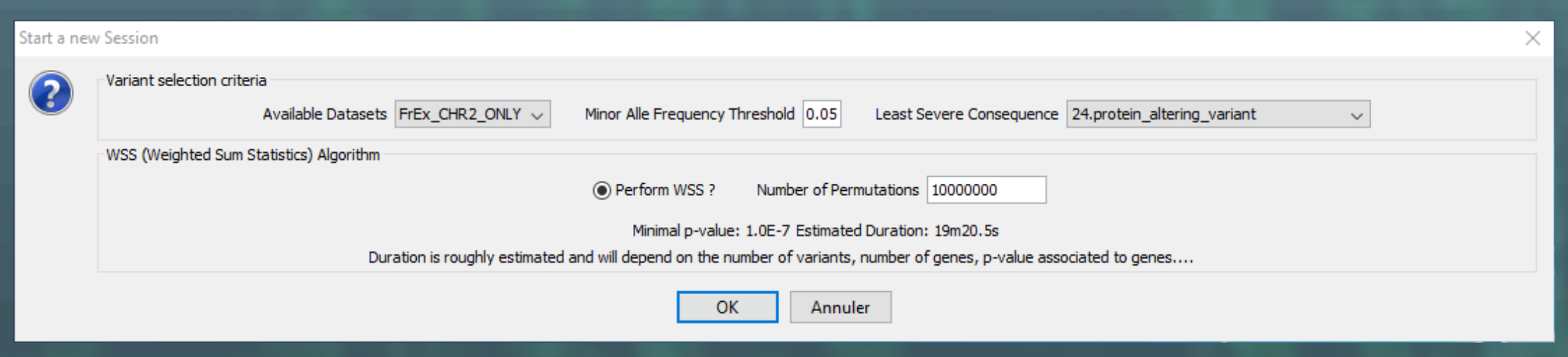
Hash Key
87D48736A170F32360D518DE4ECCDD9F25F80DE7F8A9D3FD1D1EAB0202A98DB9

Third Party Public Key
```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAmuryXQ8ZctSpr7
RL0LzHX4Cx+081DpMSP2I7ue63mH9j+m/2o4X3uVRhGHXdmAIhzCNMCpFM
5zShQrWAlx91XX4ImeJt14JAF2kb5GKolukcMwI4SAyW0/HhE+cjT3t8ye
Afal7orCK4vyVf1HNCWcZ15mSoWMsEB51c5ckCQOqN4xxxNV5AkYdiC5aF
z/DsY2UElJwEYoDYpJxf2E6/x6ODmFSHE2A2fjmFhfT087r+MWlhbuzX8Z
YmSpa0XZOvQdZp8V3qMrhnWeb9MZApeReadDZpWUHlHuMUR8YbIqa5N7bi
94YW8Q+AAZ5HArCc6Vn38qztG1iEMxRBBvkUQwIDAQAB
```

Algorithm Parameters
wss:1000000

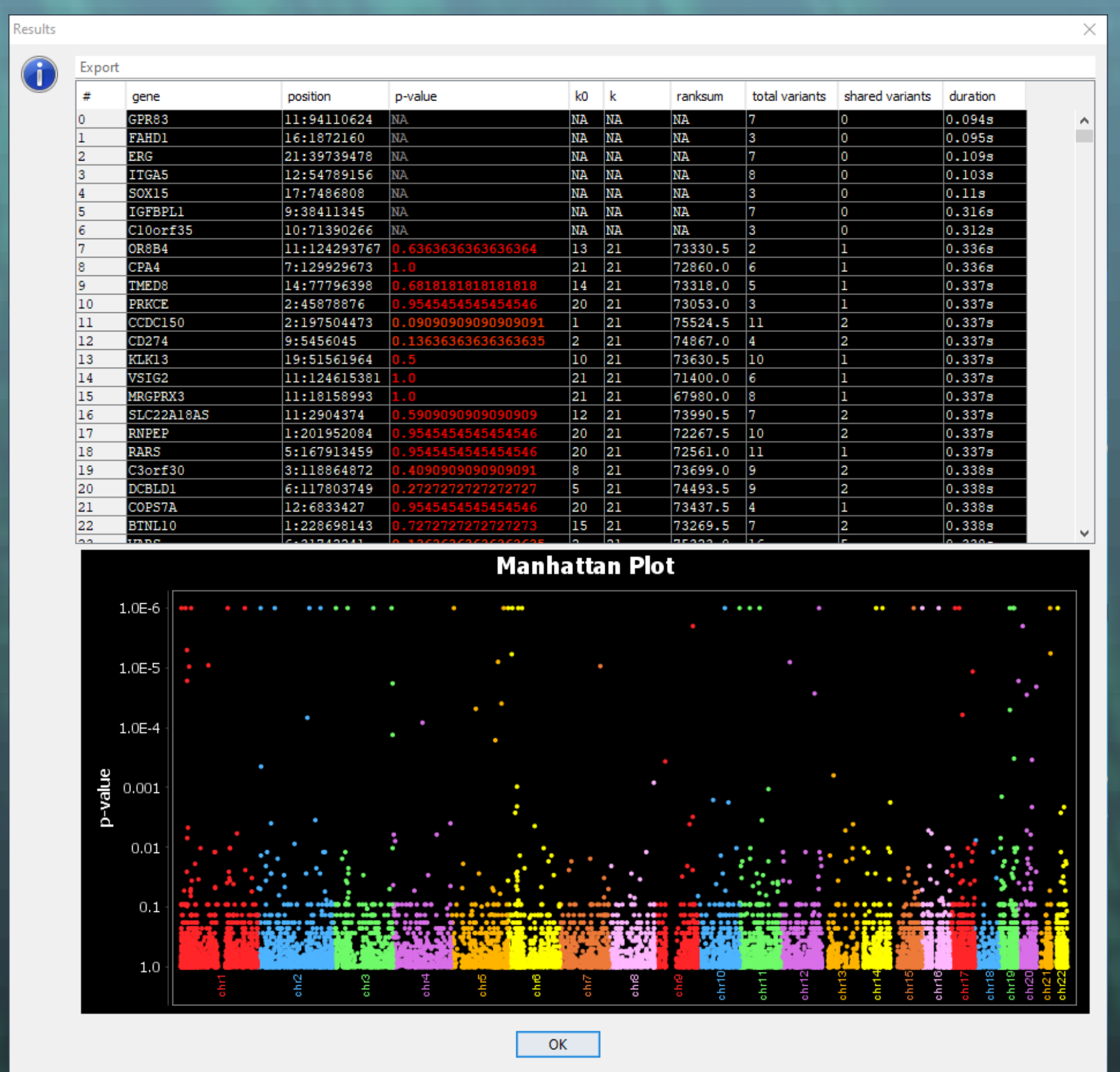
Last Known Status
[RPP 2019.06.21 11:53:35] Results Available.

Application Log
```
[16:12:12 INFO ] Welcome to PrivAS v2019-06-25
[16:12:28 INFO ] Loading session from file [C:\Users\user\Documents\Projet\PrivGene\privas\demo.session.privas]...
[16:12:28 INFO ] Connecting to RPP [alanine.univ-brest.fr:6666]...
[16:12:28 OK   ] ...Connected to RPP [alanine.univ-brest.fr:6666]
[16:12:28 INFO ] ...Status Monitoring established for session [20190621.115252.NunB]
[16:12:28 OK   ] ...Session successfully loaded
```



Variant selection criteria
Available Datasets FrEx_CHR2_ONLY
Minor Alle Frequency Threshold 0.05
Least Severe Consequence 24.protein_altering_variant

WSS (Weighted Sum Statistics) Algorithm
● Perform WSS ?   Number of Permutations 10000000
Minimal p-value: 1.0E-7 Estimated Duration: 19m20.5s
Duration is roughly estimated and will depend on the number of variants, number of genes, p-value associated to genes....



#	gene	position	p-value	k0	k	ranksum	total variants	shared variants	duration
0	GPR83	11:94110624	NA	NA	NA	NA	7	0	0.094s
1	FAHD1	16:1872160	NA	NA	NA	NA	3	0	0.095s
2	ERG	21:39739478	NA	NA	NA	NA	7	0	0.109s
3	ITGA5	12:54789156	NA	NA	NA	NA	8	0	0.103s
4	SOX15	17:7486808	NA	NA	NA	NA	3	0	0.11s
5	IGFBPL1	9:38411345	NA	NA	NA	NA	7	0	0.316s
6	C10orf35	10:71390266	NA	NA	NA	NA	3	0	0.312s
7	OR8B4	11:124293767	0.6363636363636364	13	21	73330.5	2	1	0.336s
8	CPA4	7:129929673	1.0	21	21	72860.0	6	1	0.336s
9	TMED8	14:77796398	0.6818181818181818	14	21	73318.0	5	1	0.337s
10	PRKCE	2:45878876	0.9545454545454546	20	21	73053.0	3	1	0.337s
11	CCDC150	2:197504473	0.09090909090909091	1	21	75524.5	11	2	0.337s
12	CD274	9:5456045	0.13636363636363635	2	21	74867.0	4	2	0.337s
13	KLK13	19:51561964	0.5	10	21	73630.5	10	1	0.337s
14	VSIG2	11:124615381	1.0	21	21	71400.0	6	1	0.337s
15	MRGPRX3	11:18158993	1.0	21	21	67980.0	8	1	0.337s
16	SLC22A18AS	11:2904374	0.5909090909090909	12	21	73990.5	7	2	0.337s
17	RNPEP	1:201952084	0.9545454545454546	20	21	72267.5	10	2	0.337s
18	RARS	5:167913459	0.9545454545454546	20	21	72561.0	11	1	0.337s
19	C3orf30	3:118864872	0.4090909090909091	8	21	73699.0	9	2	0.338s
20	DCBLD1	6:117803749	0.2727272727272727	5	21	74493.5	9	2	0.338s
21	COPS7A	12:6833427	0.9545454545454546	20	21	73437.5	4	1	0.338s
22	BTNL10	1:228698143	0.7272727272727273	15	21	73269.5	7	2	0.338s

http://lysine.univ-brest.fr/privas

1 Inserm, Univ Brest, EFS, UMR 1078 GGB, F-29200 Brest
2 CHRU Brest, F-29200 Brest
3 Unité INSERM UMR 1101 LaTIM, IMT Atlantique, Brest
4 INSERM DSI - CISI, F-75013, Paris, France

This work was supported by the labex COMINLABS as part of the PrivGen project