
College of Computing, Georgia Institute of Technology
CS6238 Secure Computer Systems

Secure Distributed Data Repository (SDDR)

Project 2

David Musielewicz

6083357254

david.musielewicz [at] gatech.edu

M.S. in Information Security - DL

Chidong Xue

902885919

cxue8 [at] gatech.edu

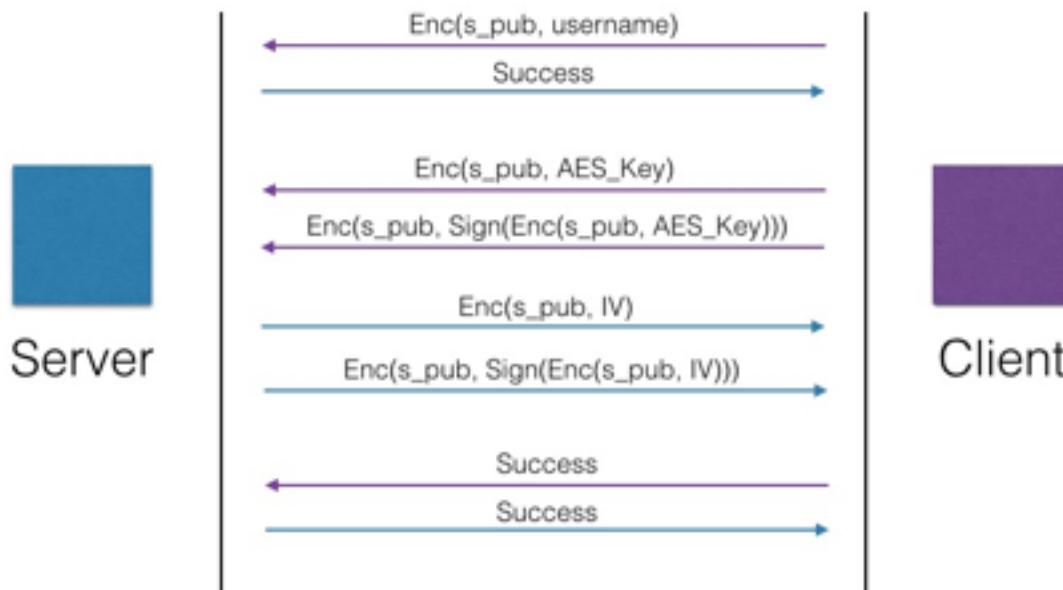
M.S. in Information Security

2014.12.02

1. Protocol Details

To implement this secure file transfer service, each node (servers and clients) all require a keystore file created using java's keytool. All keystores must contain and trust the CA certificate along with a certificate of all other trusted nodes signed by the CA.

Start-session: A client initiates a session with a server by first sending a username encrypted with the server's public key. The server verifies the username as an alias in the keystore and acknowledges receipt. The client then generates a random AES session key, encrypts it with the server's public key, signs the encrypted data, encrypts the new signature, and sends both the encrypted key and encrypted signature to the server. The server validates the signature to verify the client is indeed who they claim to be. If valid, the server generates a random IV and encrypts it using the client's public key. The server then generates a signature of the encrypted IV, encrypts the signature, and sends both the encrypted IV and signature back to the client. The client then verifies the server signature. If all signatures are valid, the connection is established and future communication is encrypted using symmetric encryption with the established session key. Below is a depiction of this interaction:



Delegate: When a client specifies that a delegation token is to be created, the request and necessary details are sent to the server. The server checks if the client owns the file being delegated by parsing the Metadata object list. If the user owns the file, a delegation token is created. This is based on the fact that the client was validated during the initial session creation,

and therefore this is a valid request. Below is the format used for the tokens stored locally by the server:

<UID>:<Client>:<Expire Date>:<Propagation>:<Access>

If the client is not the current file owner, the server parses all delegation tokens for a valid propagation token. If the server locates a token with propagation rights for this user and UID, a new token is generated; otherwise, the client receives an access denied.

Get: A client first sends a get request to the server specifying the filename or UID. The server locates files owned by other users by UID and files owned by the client by filename. If the user does not own the file, the server parses the list containing all delegation tokens looking for a token allowing the client to view the file. The token must be for the UID and requesting client, have 'get' or 'both' access, and still be within the valid time window. If the user is the owner of the file or there is a valid delegation tokens, the server transfers the file to the client. The request is denied otherwise.

Put: A client is able to put a file on the server using the security flags none, integrity, and confidential. A client sends the server the filename and desired security mode. The server generates a unique identifier (UID) by using SHA1 to hash the client's username appended with the filename. This ensures that each file is unique for each user. If a client wishes to overwrite another client's file, they must specify the UID instead of the filename. The server checks for a proper delegation token in its delegation list. The token must be for the UID and client, have 'put' or 'both' access, and not be expired. The server then adds or updates the file's information as a metadata object to an internal list. The metadata consists of the following structure:

<UID>:<Filename>:<Owner>:<Security Mode>:<Security Parameter>

The security parameter is a null value if the security mode is 'none', the file's signature if the mode is 'integrity', or an AES key encrypted with the server's private key if 'confidential'. Finally, the file is stored to the local disk as the UID and (potentially) encrypted with the generated AES key.

End-session: The client stores a SHA1 hash of each file together with the file's security mode each time the it receives a file from the server. The application then checks all received files' hash values on session termination to see if any have been modified. Copies of all altered files are then sent to the server before ending the session. The server updates these files in the same secure mode as before.

2. Evaluation

We ran PMD and FindBugs plugins in Eclipse against our application to ensure there were no security flaws in the source or byte code. FindBugs had no findings while PMD reported a significant number of violations. After parsing these violations we corrected ones related to design and security flaws that may cause potential issues, and ignored violations related to syntax and coding style.

To evaluate the overhead of security we performed multiple timing tests of all functions as well as evaluating the required overhead in source code and JAR files. Upon completion of the secure application, we duplicated our program and removed all aspects of security. This included removal of all communication encryption, integrity and confidentiality checks, and delegation functionality. This resulting application only provided basic get/put functionality and allowed users to perform functions on any requested file. All timing tests were conducted by running the server and client on the localhost to reduce timing variance in routing devices. Each application was given a one gigabyte memory pool and all files were removed from the server prior to testing.

In the end the basic application (both server and client code) was 516 lines of code, versus 1,247 lines for the original project. The compiled client and server JAR files were reduced from 46K collectively to 11K; nearly a 77% size reduction. Although a difference of 34K may seem insignificant, most applications are significantly bigger than this project. If we assume the relationship between security and application size is a linear relationship, we see that designing security into the application produces a significant amount of overhead in file size and resources required.

In addition to performing a static comparison of the applications, we performed multiple timing tests during runtime as well. We performed two sets of timing tests; one testing the security overhead added by function implementation, and one specifically evaluating the overhead in file transfer. Below are our results:

	Basic Application	Secure Application
Session Initialization	19 ms	150 ms

Secure Application:

	1KB	1MB	5MB	50MB
Put - None	2 ms	37 ms	141 ms	1606 ms
Put - Confidential	5 ms	47 ms	205 ms	2151 ms
Put - Integrity	10 ms	56 ms	188 ms	1726 ms
Get - None	6 ms	35 ms	152 ms	1450 ms
Get - Confidential	8 ms	43 ms	198 ms	1939 ms
Get - Integrity	15 ms	63 ms	215 ms	2039 ms

As expected, we see that storing a file without any secure parameters requires the least amount of time to complete. Interestingly, using the put command with the integrity flag takes the longest time to complete for small files, but is overtaken by confidentiality as the file size grows beyond 1 megabyte. This are not the same results when executing a get command for a file. We see that instead, integrity takes longer than confidentiality as files grow larger. This could be a result of our decryption function taking less time than our encryption. Overall, adding security features to these files adds an overhead that diminishes as file size gets larger. We see the largest discrepancy between timings for 1 KB files; one instance taking five times as long. As file size grows to 50 MB we see this gap reduce to only a 25% overhead. There is a slight overhead in the 'get' function because the server must first perform an access check to determine if the user has access to the file.

Basic Application:

	1KB	1MB	5MB	50MB
Put	1 ms	9 ms	65 ms	321 ms
Get	1 ms	17 ms	86 ms	443 ms

Comparing the results of our basic application to our secure one, we see that adding security throughout the project results in five times the execution time. This is a result of the access checks and network encryption overhead added by security.

After performing timing comparisons between our secure and basic application it is clear that adding security throughout an application results in significant overhead overall. This must be a strong consideration when developers are designing applications for consumer use. While security is important, its implementation should be optimized to prevent and reduce unnecessary overhead as much as possible.

3. Limitations

We were able to successfully implement all required features. However, we decided to implement the server and client as single-threaded processes. This results in only one client being able to connect to the server at any given time. This could lead to potential scaling issues if this application were implemented in a corporate environment and expected to receive concurrent traffic. Additionally, this design forces the server to act as a broker between clients when delegating tokens. However, this model also allows for strict network controls that restrict clients from needing to communicate with each other across network boundaries.

4. Individual Contributions

Both team members were part of the initial design for all features. David was responsible for the initial framework and coding structure for all functions. This included user input sanitation, and proper logic flow. David also implemented all communication between nodes. Chidong programmed and ensured that all files were properly encrypted/decrypted and updated upon session termination, and he performed all security checks and performance evaluations of the application.

Appendix: Examples

1. Login and Start-session

Client side:

```
run -- java -- 139x22
Last login: Wed Dec 3 14:04:01 on ttys002
Rick-X:run xuerick$ java -jar SDDR_Client.jar
Username:client
Password:client
SDDR>start-session(localhost)
Connection Established
SDDR>help

Secure Distributed Document Repository (SDDR)

Commands:
start-session(hostname)          - A new secure session is started with the server running at host hostname.
get(DocumentUID)                 - Request a document over the secure channel from the server.
put(DocumentUID, SecurityFlag)   - Send a document to the server over the secure channel.
delegate(DocumentUID, Client, Time, PropagationFlag, Access) - A delegation credential (e.g., signed token) is generated that allows an owner client to delegate rights (put, get or both) for a document to another client C for a time duration of T.
end-session()                    - Terminates the current session.

SDDR>|
```

Server side:

```
run -- java -- 139x22
Last login: Wed Dec 3 14:01:39 on ttys004
Rick-X:run xuerick$ java -jar SDDR_Server.jar
Waiting For Connections...
/127.0.0.1:57725: Received Username - client
/127.0.0.1:57725: Received - F05F158852C817A281EAFB8EC43AE514
/127.0.0.1:57725: Received - 680022E241D801CFA7297950E73A9F6575538F1305894727A78A25908815B86E5A0E4509A45A507094769871A940F2E554C7C56C496DE0
2AD92DF502067805989819850F56370094002A0F25908C1E5A5CB436EE0778EA235E98F4702029972157A0729608F052ACA21B53004603E644CA006342B68C8662E24E67351
0883C784380E8B8495F60FD01E0130553818A9EA8852D9E6A300E51FC8FA8AD1DA366E6050A80083D116A8F0FE7CF3AD261F6636123FDF435801A4154862F0F824C9F1429B
24E254EF06824F079E1721D02D4039015E0EE6FF11E58E9010F8C8F660857083BA453E7C81F883807918F8A5788E61AE04788178D7750E7E537738AC3FD
/127.0.0.1:57725: Sent - DC2AD4380308D187C155708F8169AF67
/127.0.0.1:57725: Sent - 2A7648517721EC9908A1EF4D8884264A9F1F2581C1498E9AE907EE838E9268827038C8E8295436E2FCFA426AE6C8871A1AE1B3CFA971CEDFEE
A7ED07707FF82283A40B00685086CB2AE83342ED4289574088E3A21CED018F8E0584A870213237748346FA3CF47409E456E703A2C9F3147EB20977F7F566E4F02572E86AC449C
2B651AE8A3EF2AA3A5F55A8C18116FAB61FC358CE73786A7AAB8C2D0308996ED4ED1AA199CA29DC96891AD5C1A160478FB8826576824A842BA9C926D40095D011C462153AA2
F443A4F0EF8112083F3ECB5C3CFED70A8C05DCBAA2C25933056E64395F889E54640F283A3988CDA39C83733628B538C40AAAE340850423449589003
/127.0.0.1:57725: Connection Established
|
```


2. Put Document

Client side:

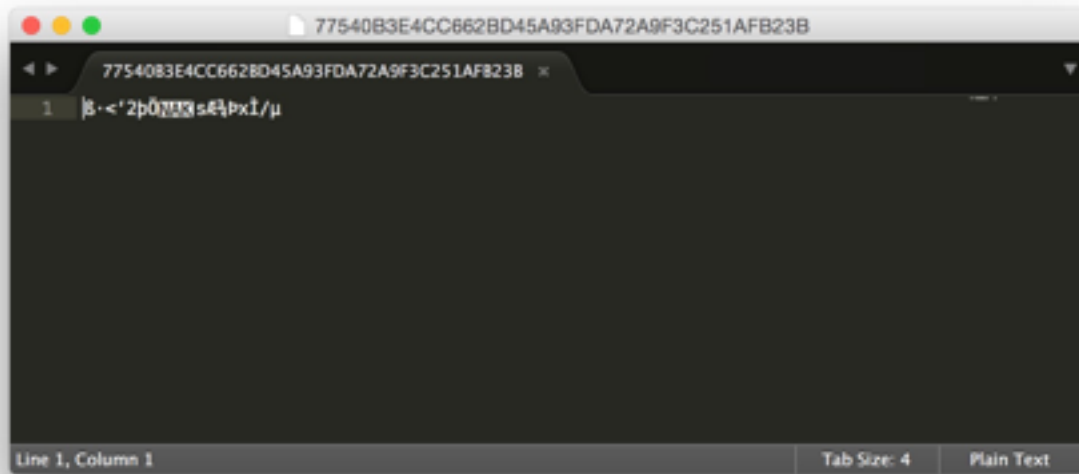
```
Commands:
start-session(hostname)           - A new secure session is started with the server running at host hostname.
get(DocumentUID)                  - Request a document over the secure channel from the server.
put(DocumentUID, SecurityFlag)    - Send a document to the server over the secure channel.
delegate(DocumentUID, Client, Time, PropagationFlag, Access) - A delegation credential (e.g., signed token) is generated that allows an owner client to delegate rights (put, get or both) for a document to another client C for a time duration of T.
end-session()                     - Terminates the current session.

S00R>put(test1, none)
New file uploaded successfully
S00R>put(test2, integrity)
New file uploaded successfully
S00R>put(test3, confidential)
New file uploaded successfully
S00R>put(test4, none)
New file uploaded successfully
S00R>|
```

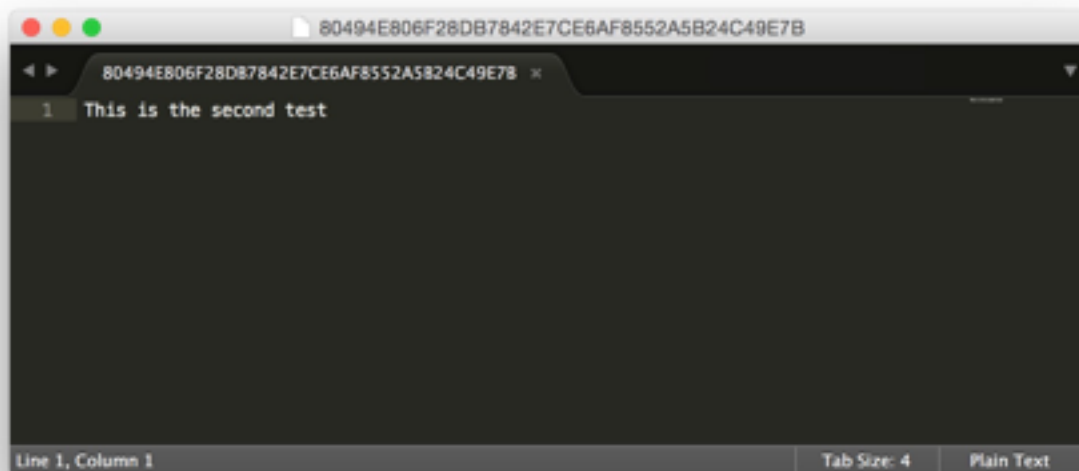
Server side:

```
/127.0.0.1:57725: Connection Established
/127.0.0.1:57725: Received - put(test1, none)
/127.0.0.1:57725: Stored - 565872A8E2A8C8C2D28FEF7A41468CF15347:test1:client:0:null
Metadata:
565872A8E2A8C8C2D28FEF7A41468CF15347:test1:client:0:null
/127.0.0.1:57725: Received - put(test2, integrity)
/127.0.0.1:57725: Stored - 80494E80F28087842E7CE6AF8552A5824C49C7B:test2:client:2:85F418E7C4538308AE6488C71E366C3DF1964221E86D8259322580158021A8D0F87A8728E476F820
A7A8CC5832385A5922674897F7F78(C5D92880588E3126164C240E601326995840D762F487A5819C64A39E4A685E99219425818C8C80A958797F898D88E878227611FF715D
9A9760888619E805093392844D70805734A82CA2F52644A858369759A51E55588156E7891ADF3A8E58318CC0E3DA5D687A3E47BAD4988852134E6134E38F2898106841D6
87281D2C295E59C330C081A3DAFE172474F0DA8E851181FA87B794A2A322D913089A295959888A7688F9A6FF8C7849491286372C07085911
182E3230C8124E2653918129967ADCA654FC286
Metadata:
565872A8E2A8C8C2D28FEF7A41468CF15347:test1:client:0:null
80494E80F28087842E7CE6AF8552A5824C49C7B:test2:client:2:85F418E7C4538308AE6488C71E366C3DF1964221E86D8259322580158021A8D0F87A8728E476F820
A7A8CC5832385A5922674897F7F78(C5D92880588E3126164C240E601326995840D762F487A5819C64A39E4A685E99219425818C8C80A958797F898D88E878227611FF715D
9A9760888619E805093392844D70805734A82CA2F52644A858369759A51E55588156E7891ADF3A8E58318CC0E3DA5D687A3E47BAD4988852134E6134E38F2898106841D6
87281D2C295E59C330C081A3DAFE172474F0DA8E851181FA87B794A2A322D913089A295959888A7688F9A6FF8C7849491286372C07085911
182E3230C8124E2653918129967ADCA654FC286
/127.0.0.1:57725: Received - put(test3, confidential)
/127.0.0.1:57725: Stored - 775A8B3E4CC6628D45A93FDA72A9F3C251AF823B:test3:client:1:8A280F594888D177D8FC8D89D88F8F587644A137A2AFD43F44C27A
60CC40986664481A9430C830F87543082AF2C5582880D27892F2509614AAAEEDE8BF291821E02E87891891290A821E87448A28E895A2A21D1936866E18C7C271F28F1A42A4F95847AFC4699118C8C82116
70D4C8802F561E0887E38DA12CA01AB71F88E318ED4E73778631930E2D9268E87B49982CAC7F6A47C15A644688C8E32987184E7191A36722894154513C1DA29C12
938F8C58892201864E08D10680F187363625C1E7606C5833831238D4772286E821E9888AC7F1E7824981226A45686D688826FC818C8836F51EC794A1FC5465
879FF3FE3005
Metadata:
565872A8E2A8C8C2D28FEF7A41468CF15347:test1:client:0:null
80494E80F28087842E7CE6AF8552A5824C49C7B:test2:client:2:85F418E7C4538308AE6488C71E366C3DF1964221E86D8259322580158021A8D0F87A8728E476F820
A7A8CC5832385A5922674897F7F78(C5D92880588E3126164C240E601326995840D762F487A5819C64A39E4A685E99219425818C8C80A958797F898D88E878227611FF715D
9A9760888619E805093392844D70805734A82CA2F52644A858369759A51E55588156E7891ADF3A8E58318CC0E3DA5D687A3E47BAD4988852134E6134E38F2898106841D6
87281D2C295E59C330C081A3DAFE172474F0DA8E851181FA87B794A2A322D913089A295959888A7688F9A6FF8C7849491286372C07085911
182E3230C8124E2653918129967ADCA654FC286
775A8B3E4CC6628D45A93FDA72A9F3C251AF823B:test3:client:1:8A280F594888D177D8FC8D89D88F8F587644A137A2AFD43F44C27A60CC40986664481A9430C830F87
543082AF2C5582880D27892F2509614AAAEEDE8BF291821E02E87891891290A821E87448A28E895A2A21D1936866E18C7C271F28F1A42A4F95847AFC4699118C8C82116
70D4C8802F561E0887E38DA12CA01AB71F88E318ED4E73778631930E2D9268E87B49982CAC7F6A47C15A644688C8E32987184E7191A36722894154513C1DA29C12
938F8C58892201864E08D10680F187363625C1E7606C5833831238D4772286E821E9888AC7F1E7824981226A45686D688826FC818C8836F51EC794A1FC5465
879FF3FE3005
/127.0.0.1:57725: Received - put(test4, none)
/127.0.0.1:57725: Stored - C7CAF8233C0DFC170E3FC8BA3CEB69E887F0488:test4:client:0:null
Metadata:
565872A8E2A8C8C2D28FEF7A41468CF15347:test1:client:0:null
80494E80F28087842E7CE6AF8552A5824C49C7B:test2:client:2:85F418E7C4538308AE6488C71E366C3DF1964221E86D8259322580158021A8D0F87A8728E476F820
A7A8CC5832385A5922674897F7F78(C5D92880588E3126164C240E601326995840D762F487A5819C64A39E4A685E99219425818C8C80A958797F898D88E878227611FF715D
9A9760888619E805093392844D70805734A82CA2F52644A858369759A51E55588156E7891ADF3A8E58318CC0E3DA5D687A3E47BAD4988852134E6134E38F2898106841D6
87281D2C295E59C330C081A3DAFE172474F0DA8E851181FA87B794A2A322D913089A295959888A7688F9A6FF8C7849491286372C07085911
182E3230C8124E2653918129967ADCA654FC286
775A8B3E4CC6628D45A93FDA72A9F3C251AF823B:test3:client:1:8A280F594888D177D8FC8D89D88F8F587644A137A2AFD43F44C27A60CC40986664481A9430C830F87
543082AF2C5582880D27892F2509614AAAEEDE8BF291821E02E87891891290A821E87448A28E895A2A21D1936866E18C7C271F28F1A42A4F95847AFC4699118C8C82116
70D4C8802F561E0887E38DA12CA01AB71F88E318ED4E73778631930E2D9268E87B49982CAC7F6A47C15A644688C8E32987184E7191A36722894154513C1DA29C12
938F8C58892201864E08D10680F187363625C1E7606C5833831238D4772286E821E9888AC7F1E7824981226A45686D688826FC818C8836F51EC794A1FC5465
879FF3FE3005
C7CAF8233C0DFC170E3FC8BA3CEB69E887F0488:test4:client:0:null
|
```


—> File test3 is stored as 77540B3E4CC662BD45A93FDA72A9F3C251AFB23B in server, with confidential mode:



—> File test2 is stored as 80494E806F28DB7842E7CE6AF8552A5B24C49E7B in server, with integrity mode: (integrity check on this file is demonstrated later in Get Document)



3. Get Document

Client side:

```
run -- java -- 129x17
New file uploaded successfully
S00R:put(test3, confidential)
New file uploaded successfully
S00R:put(test4, none)
New file uploaded successfully
S00R:get(test1)
Sent: get(test1)
File downloaded successfully
S00R:get(test2)
Sent: get(test2)
File downloaded successfully
S00R:get(test3)
Sent: get(test3)
File downloaded successfully
S00R:get(test4)
Sent: get(test4)
File downloaded successfully
```

Server side:

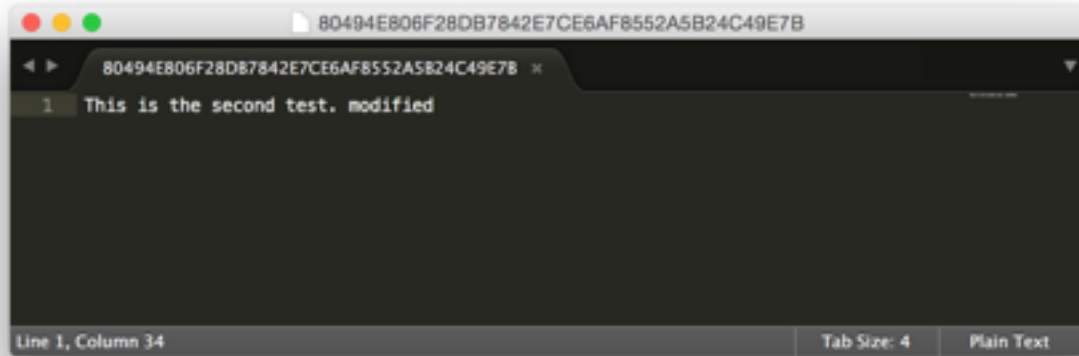
```
run -- java -- 129x20
565872A88E2A8ECBC2ED28FEF7M146BCF15347:test1:client:0:null
80494E806F28087842E7C16AF8552A5824C49E7B:test2:client:2:85F418E7CA538308A1648BC71E366C3DF1964221E368D82E593255803E5821A8D0F87A8728E476F820
A7A8CC5832385A5922674897F7F8(C5092888588E3126064C240E60132699584D0762F487A5819C64A)9E4A685E99219420818C8CA0A8587971898D88E878227611FF1E3D
9A978688619C8C509339284078985734A82CA2F52644A858369759A51E55580156E7891A2FA38E58318CC8E3DA50687A3E478AD4988852134E6134E38F28982E86841D6
8728D2C295E59C330C81A3DAFE172474F0A8E851181FA87B794A2A322D933809A95955888EA7688F9A6FF8CF854932B6372C17085911182E5230C8124E2653958129967
ADCA594FC286
775A883E4CC6628D45A93F0A72A9F3C251AF8238:test3:client:1:8A280F594880107708FC85D89DA88F5F587644A537A2AFD43F44C27A08CC40986664481A3430C830F87
5436882AF2C558288027892F29096344AAEEDE8F291621ED287891891290A821E8744BA78E89DA2A12D133686AE18C7C271F20F1A424F95847AFC4689118C8C82116
7004C8802F5611D887E381DA12CA0EAB71F88E314ED4E737786339304E20F268E8E78A99E2CAAC7F6AA7C15A644688CAB332987184821E93A36722994154513C1DA2FC12
938F8CF58892201864E08D1066D8F187363625C1E7680C5833E3E123ED84772280E521E9988ACF71E782498E1226A435886D6888E26FC8638C8836F51EC72F4A2FC5465
879FF3FE3005
C7CAF8233CEDFCL7DE5FC8BA3CEE869E887F0488:test4:client:0:null
/127.0.0.1:57725: Received - get(test1)
/127.0.0.1:57725: Sent File - 565872A88E2A8ECBC2ED28FEF7M146BCF15347
/127.0.0.1:57725: Received - get(test2)
/127.0.0.1:57725: Sent File - 80494E806F28087842E7C16AF8552A5824C49E7B
/127.0.0.1:57725: Received - get(test3)
/127.0.0.1:57725: Sent File - 775A883E4CC6628D45A93F0A72A9F3C251AF8238
/127.0.0.1:57725: Received - get(test4)
/127.0.0.1:57725: Sent File - C7CAF8233CEDFCL7DE5FC8BA3CEE869E887F0488
```

—> The confidential file test3 is decrypted and saved locally:

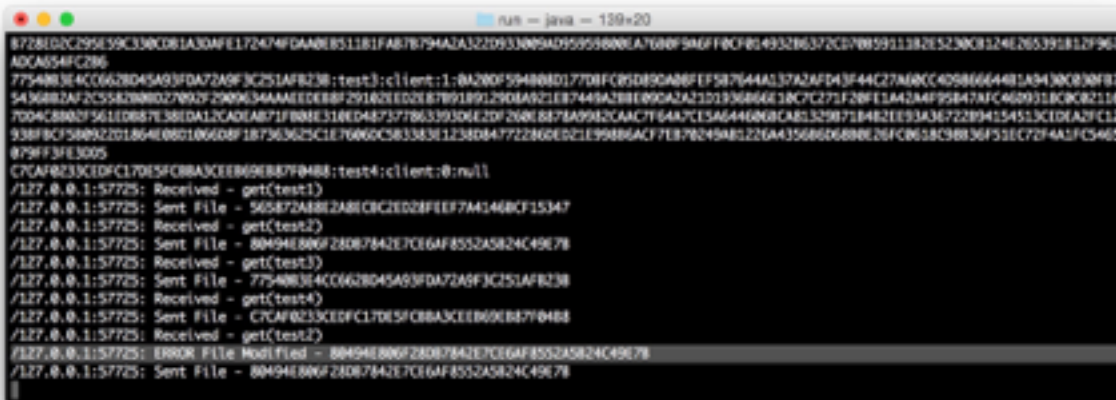
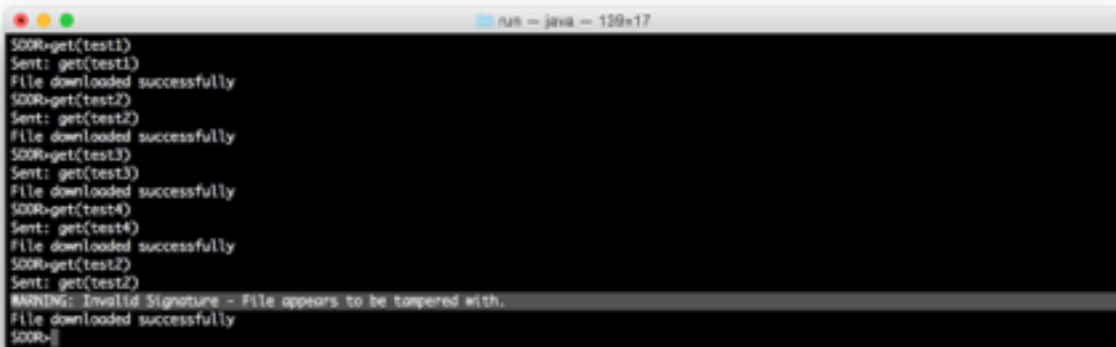
```
test3
1 |test number 3!!!-
```

—> For the integrity file test2, we first modify it on the server:

(modify 80494E806F28DB7842E7CE6AF8552A5B24C49E7B on the server)



Then, try to retrieve the file again. System gives a warning to the client that it has been modified:



4. Delegation

4.1 Grant delegation

Client side:

```
run — bash — 103x17
SDOR>delegate(test1, client2, 10000, true, both)
Delegation successful
SDOR>delegate(test2, client2, 10000, true, get)
Delegation successful
SDOR>delegate(test3, client2, 10000, false, put)
Delegation successful
SDOR>delegate(test4, all, 10000, false, put)
Delegation successful
SDOR>delegate(test5, client2, 1, true, get)
Delegation successful
SDOR>end-session
Sent: end-session
Session Ended
SDOR>exit
No session established
Bye!
Rick-X:run xuerick$
```

Server side:

```
run — java — 103x27
/127.0.0.1:57864: Received - delegate(test1, client2, 10000, true, both)
Delegations:
565872A88E2A8ECBC2ED28FEEF7A4146BCF15347:client2:1417645708614:true:0
/127.0.0.1:57864: Received - delegate(test2, client2, 10000, true, get)
Delegations:
565872A88E2A8ECBC2ED28FEEF7A4146BCF15347:client2:1417645708614:true:0
80494E806F28D87842E7CE6AF8552A5B24C49E7B:client2:1417645711753:true:1
/127.0.0.1:57864: Received - delegate(test3, client2, 10000, false, put)
Delegations:
565872A88E2A8ECBC2ED28FEEF7A4146BCF15347:client2:1417645708614:true:0
80494E806F28D87842E7CE6AF8552A5B24C49E7B:client2:1417645711753:true:1
7754083E4CC662BD45A93FDA72A9F3C251AFB23B:client2:1417645718014:false:2
/127.0.0.1:57864: Received - delegate(test4, all, 10000, false, put)
Delegations:
565872A88E2A8ECBC2ED28FEEF7A4146BCF15347:client2:1417645708614:true:0
80494E806F28D87842E7CE6AF8552A5B24C49E7B:client2:1417645711753:true:1
7754083E4CC662BD45A93FDA72A9F3C251AFB23B:client2:1417645718014:false:2
C7CAF0233CEDFC17DE5FCBBA3CEEB69EB87F0488:all:1417645725503:false:2
/127.0.0.1:57864: Received - delegate(test5, client2, 1, true, get)
Delegations:
565872A88E2A8ECBC2ED28FEEF7A4146BCF15347:client2:1417645708614:true:0
80494E806F28D87842E7CE6AF8552A5B24C49E7B:client2:1417645711753:true:1
7754083E4CC662BD45A93FDA72A9F3C251AFB23B:client2:1417645718014:false:2
C7CAF0233CEDFC17DE5FCBBA3CEEB69EB87F0488:all:1417645725503:false:2
FAS456882457803DA0819089B0C5CC57D0324641:client2:1417635731239:true:1
/127.0.0.1:57864: Received - end-session
```


4.2 Performing delegation

```
run -- java -- 103x29
Username:client2
Password:client2
SDDR>start-session(localhost)
Connection Established
SDDR>get(565872A88E2A8ECBC2ED28FEEF7A41468CF15347)
Sent: get(565872A88E2A8ECBC2ED28FEEF7A41468CF15347)
File downloaded successfully
SDDR>get(80494E806F28D87842E7CE6AF8552A5824C49E7B)
Sent: get(80494E806F28D87842E7CE6AF8552A5824C49E7B)
File downloaded successfully
SDDR>get(7754083E4CC662BD45A93FDA72A9F3C251AFB23B)
Sent: get(7754083E4CC662BD45A93FDA72A9F3C251AFB23B)
File download failed
SDDR>get(FA5456882457803DAD81908980C5CC57D0324641)
Sent: get(FA5456882457803DAD81908980C5CC57D0324641)
File download failed
SDDR>put(565872A88E2A8ECBC2ED28FEEF7A41468CF15347, integrity)
File updated successfully
SDDR>put(80494E806F28D87842E7CE6AF8552A5824C49E7B, confidential)
File upload failed
SDDR>delegate(test1, client3, 1000, true, both)
Delegation successful
SDDR>delegate(test2, client3, 1000, false, get)
Delegation successful
SDDR>delegate(test2, client3, 1000, false, put)
ERROR Access Denied
SDDR>delegate(test3, client3, 1000, false, both)
ERROR Access Denied
SDDR>
```

```
run -- java -- 189x52
/127.0.0.1:5780: Received - get(565872A88E2A8ECBC2ED28FEEF7A41468CF15347)
/127.0.0.1:5780: Sent File - 565872A88E2A8ECBC2ED28FEEF7A41468CF15347
/127.0.0.1:5780: Received - get(80494E806F28D87842E7CE6AF8552A5824C49E7B)
/127.0.0.1:5780: Sent File - 80494E806F28D87842E7CE6AF8552A5824C49E7B
/127.0.0.1:5780: Received - get(7754083E4CC662BD45A93FDA72A9F3C251AFB23B)
/127.0.0.1:5780: ERROR Denied Access - 7754083E4CC662BD45A93FDA72A9F3C251AFB23B
/127.0.0.1:5780: Received - get(FA5456882457803DAD81908980C5CC57D0324641)
/127.0.0.1:5780: ERROR Denied Access - FA5456882457803DAD81908980C5CC57D0324641
/127.0.0.1:5780: Received - put(565872A88E2A8ECBC2ED28FEEF7A41468CF15347, integrity)
/127.0.0.1:5780: Stored - 565872A88E2A8ECBC2ED28FEEF7A41468CF15347: test1: client: 2: 70626917BF608E8F552830580A5F2117B21205912A8075061F38E5C1C469F4097F4E80F7501C003C1FABCA389E087E94A48585
1E07972C1498817406F8E78E2300224AC75454893850F4881376508E0BF5A424232C0888C30A16E798802E85031E056A62F1685CF781F8C8B98FDB350383C362E8FA7684F3CC152445C218054E1F98837886AF69F3314434140490
000E3134821C831EA28544888523495A450385A867F7CC868E2867208054301ED039566A8224FAF63267C5E62043FB702D0623F7C8B30AF874877FEC21148A832DA50C61612A1FDFC088D5F5A87E3C88BC2908148A22C35DF84C4
63881A8F8FED30C880999464952
Metadata:
565872A88E2A8ECBC2ED28FEEF7A41468CF15347: test1: client: 2: 70626917BF608E8F552830580A5F2117B21205912A8075061F38E5C1C469F4097F4E80F7501C003C1FABCA389E087E94A485851E07972C1498817406F8E78E230
0224AC75454893850F4881376508E0BF5A424232C0888C30A16E798802E85031E056A62F1685CF781F8C8B98FDB350383C362E8FA7684F3CC152445C218054E1F98837886AF69F3314434140490000E3134821C831EA28544888523495A450385A867F7CC868E2867208054301ED039566A8224FAF63267C5E62043FB702D0623F7C8B30AF874877FEC21148A832DA50C61612A1FDFC088D5F5A87E3C88BC2908148A22C35DF84C463881A8F8FED30C880999464952
7
80494E806F28D87842E7CE6AF8552A5824C49E7B: test2: client: 2: 746074C7A75613908284C7B36E23113C0649F3A8055113091C82388E7084CF60E0DE880736267F168871A5963EAB340665E8998C1A3E507D0655CA836838
F0887085C80C3538A025083AA13DCEFC5C8315F0303A8214651188631A8812097770E898982A5E079CC8617E95A8883921971CF80CE71FF7C0368A7E1246C23508953E8AB340C350F258A6CE24FAA5FF3888938C085A686712
13495A450385A867F7CC868E2867208054301ED039566A8224FAF63267C5E62043FB702D0623F7C8B30AF874877FEC21148A832DA50C61612A1FDFC088D5F5A87E3C88BC2908148A22C35DF84C463881A8F8FED30C880999464952
7
7754083E4CC662BD45A93FDA72A9F3C251AFB23B: test3: client: 1: 7062798F5C4D29680684470A2A338E6648880F8667D6A3FDB8A80G81232A3E7506F818C3F54720C30A67904A76715116FCAC1787CA888907830AE2E27524
8342CF788A5A12CE3452647672068091EA72381D1278C18980353F2A25E368EA94A119FC5FC31A154346FAA5F93812899031A8A7C70907E8854CC0F5A2D855AF69C3548CE7E736A82A8E6A8A6G51A792901A5AB2AC19391BF848CD
5CF1AF8C2B79358262CB408F3E466662E86407FD251CF24E30A538AF25A15966CA521BAE84A81848F83E4144863BF6A608375C45F60388C1965955F07E8748C8E5F6981C4CA440900276AE658CFED40C15086743843D498E4598C26
1
C7CAF823C10FC17DE5FC8A3CEE869887F8488: test4: client: 0: null
FA5456882457803DAD81908980C5CC57D0324641: test5: client: 0: null
/127.0.0.1:5780: Received - put(80494E806F28D87842E7CE6AF8552A5824C49E7B, confidential)
/127.0.0.1:5780: ERROR Denied Access - 80494E806F28D87842E7CE6AF8552A5824C49E7B: test2: client: 2: 746074C7A75613908284C7B36E23113C0649F3A8055113091C82388E7084CF60E0DE880736267F168871A5963EAB340665E8998C1A3E507D0655CA836838
F0887085C80C3538A025083AA13DCEFC5C8315F0303A8214651188631A8812097770E898982A5E079CC8617E95A8883921971CF80CE71FF7C0368A7E1246C23508953E8AB340C350F258A6CE24FAA5FF3888938C085A686712
13495A450385A867F7CC868E2867208054301ED039566A8224FAF63267C5E62043FB702D0623F7C8B30AF874877FEC21148A832DA50C61612A1FDFC088D5F5A87E3C88BC2908148A22C35DF84C463881A8F8FED30C880999464952
7
/127.0.0.1:5780: Received - delegate(test1, client3, 1000, true, both)
Delegations:
565872A88E2A8ECBC2ED28FEEF7A41468CF15347: client2: 1417645708614: true: 0
80494E806F28D87842E7CE6AF8552A5824C49E7B: client2: 1417645711753: true: 1
7754083E4CC662BD45A93FDA72A9F3C251AFB23B: client2: 1417645718014: false: 2
C7CAF823C10FC17DE5FC8A3CEE869887F8488: all: 1417645725983: false: 2
FA5456882457803DAD81908980C5CC57D0324641: client2: 1417635731239: true: 1
565872A88E2A8ECBC2ED28FEEF7A41468CF15347: client3: 1417636456190: true: 0
/127.0.0.1:5780: Received - delegate(test2, client3, 1000, false, get)
Delegations:
565872A88E2A8ECBC2ED28FEEF7A41468CF15347: client2: 1417645708614: true: 0
80494E806F28D87842E7CE6AF8552A5824C49E7B: client2: 1417645711753: true: 1
7754083E4CC662BD45A93FDA72A9F3C251AFB23B: client2: 1417645718014: false: 2
C7CAF823C10FC17DE5FC8A3CEE869887F8488: all: 1417645725983: false: 2
FA5456882457803DAD81908980C5CC57D0324641: client2: 1417635731239: true: 1
565872A88E2A8ECBC2ED28FEEF7A41468CF15347: client3: 1417636456190: true: 0
80494E806F28D87842E7CE6AF8552A5824C49E7B: client3: 1417636460007: false: 1
/127.0.0.1:5780: Received - delegate(test2, client3, 1000, false, put)
/127.0.0.1:5780: Received - delegate(test3, client3, 1000, false, both)
```

—> Explanations:

After ‘client’ grants delegations, login as client2.

(1) File 565872A88E2A8ECBC2ED28FEEF7A4146BCF15347 (test1), and file 80494E806F28DB7842E7CE6AF8552A5B24C49E7B (test2) can be downloaded successfully, since client2 has ‘get’ access for these files from the file owner.

(2) File 77540B3E4CC662BD45A93FDA72A9F3C251AFB23B (test3) and file FA5456882457803DAD819089B0C5CC57D0324641 (test4) cannot be downloaded, since client2 only has ‘put’ delegation for these files.

(3) File 565872A88E2A8ECBC2ED28FEEF7A4146BCF15347 (test1) can be put by client2 while 80494E806F28DB7842E7CE6AF8552A5B24C49E7B (test2) cannot.

(4) File test1 and test2’s delegation rights can be propagated with the same delegation right from the user ‘client’. File test3’s delegation right cannot be propagated.

(5) However, file test2 cannot be propagated with the delegation right ‘put’, since client2 only has the ‘get’ delegation right from user ‘client’

5. End-session and Updating

run - java - 117x31

```
S00R>put(test2, none)
New file uploaded successfully
S00R>put(test3, none)
New file uploaded successfully
S00R>put(test4, confidential)
New file uploaded successfully
S00R>put(test5, integrity)
New file uploaded successfully
S00R>get(test2)
Sent: get(test2)
File downloaded successfully
S00R>get(test3)
Sent: get(test3)
File downloaded successfully
S00R>get(test4)
Sent: get(test4)
File downloaded successfully
S00R>get(test5)
Sent: get(test5)
File downloaded successfully
S00R>end-session()
Sent: end-session()
test3 is changed and updated accordingly.
test4 is changed and updated accordingly.
test5 is changed and updated accordingly.
Session Ended
S00R>start-session(localhost)
Connection Established
S00R>shutdown
Sent: shutdown
S00R>||
```

run - bash - 160x40

```
/127.0.0.1:57953: Received - get(test2)
/127.0.0.1:57953: Sent File - 80494E80F28D87842E7CE6AF8552A5824C49E7B
/127.0.0.1:57953: Received - get(test3)
/127.0.0.1:57953: Sent File - 7754083E4CC6628D45A03F0A72A9F3C251AF823B
/127.0.0.1:57953: Received - get(test4)
/127.0.0.1:57953: Sent File - C7CAF0233C0DFC170E5FCBBA3CEE069E87F0488
/127.0.0.1:57953: Received - get(test5)
/127.0.0.1:57953: Sent File - FA54568824578030A081908980C5CC57D0324641
/127.0.0.1:57953: Received - end-session
test3 updated
test4 updated
test5 updated
/127.0.0.1:57960: Received Username - client
/127.0.0.1:57960: Received - 273651D0C885A65889A48A6431311E
/127.0.0.1:57960: Received - 3309C2058F141F0182F1C9FA23189AC91A1E9674C094481152E503AB19567886018338581F26A71905281C619F29CF4EE3C32FE3F5900719F66A8138C086C2F788
790F45F86FC77402389F397DE5BCCADFFDF9CFECCDF824EED0635FA4506A9F0B1772179EF6C665812FA1C4674EE4985524DAE94CD285678C2ACFC7509208841E288EC267D785542E1432085588A181CF2
EED515CF7318812805A99A81772FA9AD04B37FAA9FFA8FAEDES993F693C291840E9A82522E7903630918100A833E82121CC7F57C520F8719C90283608F99001E1690EC628DE19388D948FC20E6
617851308998881F0115888038D0173FA151C53438AE79F06763667B70CA2
/127.0.0.1:57960: Sent - 0488903E6107F798EBEE6FA981413A1
/127.0.0.1:57960: Sent - 70B34CA88735CF9304CE25DF8F081EB2C80F20A804EC32C331558F118F530636CE26E918A449A8065A1C5498E3FC825A7252FEB7902CE5AF4F5E0E9F6A998990
D0CE35838BFEC967F48AF1C82EC60D12857C9382DF37DF40CED13F87BCDC6585C5CFD29C525C8524FABFCB8DC615C4333A80700762B24065094F1895C38E2AF608014A379136498E87DA0185A2F3
82D483FE6957DE6912042E7987887C8088109FC9880916351228342951DA598921748EA8A605482C7FA995055F276E188439AF1539758FE2199AC90385F73EF9077280C80E40A45868319898A90
1FEE95BA38F743CA534689250A8FC153215197831A6070197A8590579
/127.0.0.1:57960: Connection Established
/127.0.0.1:57960: Received - shutdown

Metadata file content:
565872A88E2A8E08C2ED08FEEF7A41468CF15347:test1:client:0:null
80494E80F28D87842E7CE6AF8552A5824C49E7B:test2:client:0:null
7754083E4CC6628D45A03F0A72A9F3C251AF823B:test3:client:0:null
C7CAF0233C0DFC170E5FCBBA3CEE069E87F0488:test4:client:1:3644460530FCE7596611A49FCD0E53C5217888215A5E5F08C3538D26A60C5EE2535A998B798AE9FAC20E767A831BF0E1D578E
93A3AA23892E3A3A18807D03119007F464513DDCF75E5AA8590E68985878CF35F407F483CC4DC057FF5A9C867ECFF634859069050A0C289FA2508A471A3357EE42ECB4C632A14874902078C153480
23A74089977DA25185475A8338706060E05A848D9296806426ED128F639858300A8F47D043EE3AF3CF2F4018FD43A6486C1A83F8F814C878940C80C1AAAA15E24039EE8748C8A95FA21931EC2450C780
EA957E82C7313AF4F2BAGDAD8AC88D0C3FD07AD37CF64825FDF80A8362CEB8562C137284668B0F915C5E8
FA54568824578030A081908980C5CC57D0324641:test5:client:2:607A3E5E40F37709273AF0A18D151E9581F9FAD544F76423A559858F312A60A3E805BF3A49A340ACDF30802241215855E4A2D243
4A9CF819CABAS780845663786C1A7E6C8E1E076E089351C28194418FBCA3AA1E16C485420377FEAD9C314CFD900C46EDAFCFB020593109360A636AD1210F28117E8529F6204C85F84C71FFD1872008
C7A348A76329CED32951CFD098648CA900A5898F0DEDE333E5034687743F79678C88CEC41A608F090483C188488F08E11C4ASC3FE7E15CEDE859C88150ED04689803AD062E24A8E85397DC6CAA3DC2
4ED4233671D515F2C424FC4C751D8285F9E124898C989AF23435F478FF2731E835300F0F386C8A468916891
```


—> Upload test2, test3, test4, and test5, with security mode none, none, confidential and integrity, respectively. Then download these 4 files, and modify test3, test4, and test5.

After executing `end-session()`, system updates test3, test4, and test5 to the server, with the original security mode.