Concept Activation Authentication

Your Rating: ☆☆☆☆☆ Results: ★★★☆☆ 92 rates

GREY Implementation tracked in MGNLXAA-27@jira GREY

Up until now Activation process relied on user credentials to authenticate validity of the request, this requires to keep instance users in sync and presents potential problem if user login details were compromised. Rather than relying on the users, proposed change attempts to establish credibility between author and all its public instances that depends on secure key, but not on any particular user.

current status

In order to authorize incoming activation, user or system credentials from the author instance are used to login to the public instance. This leads to few types of potential problems:

- credentials between authoring and public envirnment needs to be in sync
- any author instance with a knowledge of public instance address and same set of credentials is able to push the activation through
- activation process is succeptable to man-in-the-middle attack which can record the communication and reply later with same or different content.

Last two points above can be mitigated by allowing activation only from given single IP address.
The process can be further secured by using https to perform activation instead of plain http.

Since transfer can be secured very effectively simply by switching to https, this concept will not focus on securing the transfer, but only on providing means for public instance to authenticate that author sending the data is valid and that the transaction is not a replay of some older (de)activation.

Constrains:

- choosen mechanism needs to work for multiple subscribers (public instances).

Target

Activation mechanism clearly identifying the single author instance, not dependening on user or system credentials and making it reasonable difficult for anyone to replay operations at later time.

- include time of transaction, and signatures of all the resources being sent to public as part of the activation
- use public/private key to sign the above info
- autogenerate or distribute the author's instance public key to assigned public instance after installation or upon first activation or manually

- Including the time should protect from replaying the operation at later time.
- Including the signatures should protect the transaction from information tampering
- Encrypting both info should protect that information itself from being tampered with and at the same time it provides the means for public to authenticate sender since the information could be decrypted only with the public key associated with the private key held by the author instance.
- Compromise of any public instance would still not endanger other public instances since the public key can't be used to successfully fake the said information and fake transaction
- encrypting only the essential information instead of everything lowers the amount of resources necessary to secure the transaction.
- one way asynchronous encryption makes transfer secure, allows for unsecure distribution of public key and allows reuse of same info for distribution to all public servers involved.
- required effort to implement the above is minimal

UI components

- admin page to generate new key and send it to all currently known public instances ... either using the old key or nothing in case of new instance installation. Export of public key should be also possible.
- alert task in the activation command chain to warn user that secure communication was not yet established?

Open questions

- where to store the private key? Preferably it would not be visible to anyone and couldn't be activated.
- is alert good idea? ... Amount of work to test update tasks that change existing command chains is usually big. And i'm not convinced of the value.
- anything else?