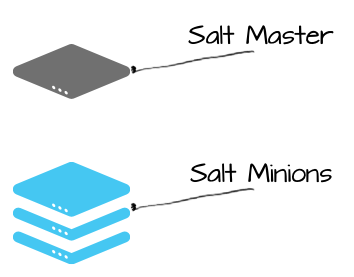
COMMUNICATION & SECURITY

ARCHITECTURE MODEL

- Salt sử dụng kiến trúc Server-agent để kết nối các thành phần

- Thành phần chính bao gồm Salt master(server) và Salt minion(agent)



- Salt master chịu trách nhiệm gửi command tới các Salt minions, và sau đó tập hợp các kết quả, hiện thị các kết quả.

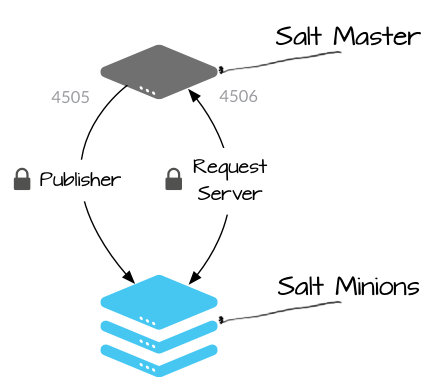
- 1 single Salt master có thể quản lý hàng ngàn minions.

COMMUNICATION MODEL

- Salt kết nối với hệ thống quản lý thông qua mô hình publish-subscribe.

- Kết nối được khởi tạo bởi Salt minion, đồng nghĩa với việc không cần mở bất kỳ port tới tại minion.

- Salt master sử dụng ports 4505 và 4506, 2 port này cần được mở để accept các kết nối tới.

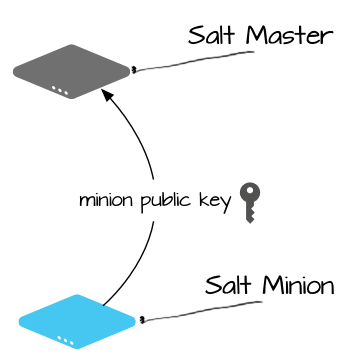


|  |  |
| --- | --- |
| Publisher | (port 4505) tất cả minions sẽ được kết nối tới cổng publisher, đây là cổng được kết nối tới để nhận các messages. Command sẽ được gửi bất đồng bộ tới tất cả kết nối thông quả port này, nó cho phép thực hiện 1 lượng lớn đồng thời. |
| Request Server | (port 4506) Salt minions kết nối tới port nhằm mục đích yều cầu các kết quả từ Salt master, bảo đảm các request file, các giá trị đặc biệt của minion (gọi là Salt pillar). Kết nối tới port này là 1:1 giữ Salt master và Salt minion (không bất đồng bộ). |

SALT MINION AUTHENTICATION

- Khi minion bắt đầu kết nối lần đầu tiên, nó sẽ tìm kiếm hệ thống có tên là salt trong mạng (có thể là ip hoặc hostname).

- Khi tìm thấy, minion sẽ khởi tại quá trình bắt tay và gửi public key tới Salt master



- Sau khi kết nối được khởi tạo, Salt minions public key sẽ được lưu trữ tại server, nó cần phải được Salt master câp nhận thông qua “salt-key” cmd.

- Đây là phần gây nhầm lẫn cho người dùng, Salt sẽ không cung cấp key giải mã thông điệp cho đến khi Salt minion public key được chấp nhận bởi Salt master.

- Sau khi minion key được chập nhận, Salt master sẽ trả lại public key cùng 1 khóa AES luôn phiên cho việc giải mã các thông điệp được gửi bởi Salt master.

- AES key sẽ được mã hóa bởi key public của Salt minion để nó có thể được giải mã tại minion.

SECURE COMMUNICATION

- Bảo đảm kết nối

- Tất cả các thông điệp giữa Salt master và Salt minion được mã hóa bởi AES keys.

Note:

The AES encryption key uses an explicit initialization vector and a CBC block-chaining algorithm in accordance with the latest accepted version of TLS.

ROTATING SECURITY KEYS

- Khóa luân phiên AES được sử dụng để mã hóa jobs được gửi từ Salt master tới Salt minions và kết nối tới Salt master fileserver. Key mới sẽ được sinh ra mỗi khi Salt master được khởi động lại và khi Salt minion key bị xóa bởi “salt-key” cmd

- Khi khóa được quay vòng, Tất cả minions sẽ được chứng thực lại để nhận key.

### Chưa dịch

ENCRYPTED COMMUNICATION CHANNELS

Publication communication between the Salt master and Salt minion is encrypted using a rotating AES key. Direct communication between the Salt master and Salt minion is encrypted using a unique AES key for each session.

For example, published jobs are encrypted using the rotating AES key, while secure, minion-specific data sent using Salt pillar is encrypted with a unique AES key for each Salt minion and session.

USER ACCESS CONTROL

Before a command is sent to minions, Salt performs some internal checks against the Publisher ACL to make sure that the user executing the command has the correct privileges. If the user is authorized to run the specified command against the specified target, the command is sent. If not, an error is returned. Salt also returns a list of the minions that are expected to respond to the command line interface, so it knows how long to wait for returns.

Next, we’ll go over the main pluggable subsystems that you’ll use to manage infrastructure, starting with remote execution.

e