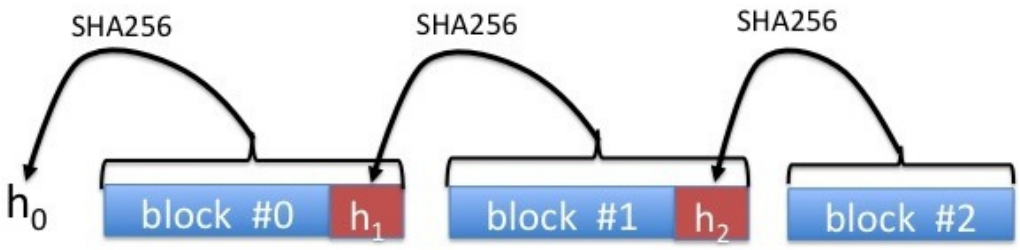
Bài tập lập trình 4[1]

Một trang web lưu trữ file video lớn F để mọi người có thể tải xuống. Các trình duyệt tải file này về cần phải đảm bảo tính xác thực của file trước khi hiển thị nội dung cho người dùng. Một cách tiếp cận là trang Web băm nội dung của F dùng hàm băm kháng xung đột H rồi gửi giá trị băm $h = H(F)$ cho người dùng thông qua một kênh truyền tin cậy. Trình duyệt sẽ tải xuống toàn bộ file F, kiểm tra xem $H(F)$ có bằng với giá trị băm xác thực h, và nếu đúng thì sẽ hiển thị video cho người dùng.

Không may, với cách làm này, video sẽ chỉ được hiển thị sau khi **toàn bộ** file F đã được tải xuống. Mục đích của bài tập này là xây dựng một hệ thống xác thực file sao cho trình duyệt có thể xác thực và hiển thị nội dung từng đoạn video ngay khi chúng được tải về chứ không phải đợi đến khi tải hết toàn bộ file.

Thay vì tính mã băm của toàn bộ file, trang web sẽ tách file thành từng khối $1KB$ (1024 byte). Nó tính mã băm của khối cuối cùng và thêm giá trị băm này vào khối thứ hai tính từ cuối. Sau đó nó tính mã băm của khối thứ hai đã được thêm mã băm này và thêm giá trị băm này vào khối thứ ba tính từ cuối. Quá trình tiếp tục từ khối cuối cùng trở về khối đầu tiên như sơ đồ dưới đây:



Hình 1: quá trình băm

Giá trị băm cuối cùng h_0, chính là mã băm của khối đầu tiên ghép với mã băm được thêm. Giá trị h_0 này được gửi tới người dùng qua một kênh tin cậy.

Bây giờ, trình duyệt tải file F, lần lượt từng khối, trong đó mỗi khối sẽ bao gồm giá trị băm được thêm như sơ đồ ở trên. Khi nhận được khối đầu tiên $(B_0 \,\|\, h_1)$, trình duyệt sẽ kiểm tra xem $H(B_0 \,\|\, h_1)$ có bằng với h_0 và nếu đúng nó bắt đầu hiển thị khối video đầu tiên. Khi nhận được khối thứ hai $(B_1 \,\|\, h_2)$, trình duyệt sẽ kiểm tra xem $(B_1 \,\|\, h_2)$ có bằng với h_1 và nếu đúng nó hiển thị khối video thứ hai. Quá trình này sẽ tiếp tục cho tới khối cuối cùng. Bằng cách này, mỗi khối sẽ được xác thực và hiển thị ngay khi nó được nhận và không cần đợi cho tới khi toàn bộ file được tải xuống.

Dễ thấy, nếu hàm băm H là kháng xung đột thì kẻ tấn công không thể sửa đổi bất kỳ khối video nào mà không bị phát hiện bởi trình duyệt. Thật vậy, vì $h_0 = H(B_0 \,\|\, h_1)$, kẻ tấn công

[1]Source: `https://class.coursera.org/crypto-015/quiz/attempt?quiz_id=122`

không thể tìm được cặp $(B'_0, h'_1) \neq (B_0, h_1)$ thoả mãn $h_0 = H(B'_0 \,\|\, h'_1)$ vì điều này vi phạm tính kháng xung đột của H. Vậy thì sau khi mã băm đầu tiên được kiểm tra, trình duyệt thấy rằng cả B_0 và h_1 đều xác thực. Cùng cách lập luận chỉ ra rằng sau khi mã băm thứ hai được kiểm tra, trình duyệt thấy rằng cả B_1 và h_2 đều xác thực, và tương tự cho các khối còn lại.

Trong bài tập này, chúng ta sẽ dùng SHA256 làm hàm băm. Bạn có thể dùng SHA256 trong thư viện mật mã sẵn có mà không cần cài đặt lại. Ví dụ, bạn có thể dùng thư viện mật mã PyCrypto (Python), hoặc Crypto++ (C++). Khi thêm giá trị băm vào mỗi khối, hãy thêm giá trị băm ở dạng nhị phân (cụ thể, dãy 32 byte = 256 bit). Nếu kích thước của file không chia hết cho $1KB$ thì khối cuối cùng sẽ ngắn hơn $1KB$ nhưng các khối khác sẽ dài đúng $1KB$.

Nhiệm vụ của bạn là viết mã để tính giá trị h_0 của một file F và kiểm tra các khối của F ngay khi nhận được.

Bạn có thể kiểm tra chương trình của bạn bằng cách dùng nó để băm các file khác nhau mà bạn tìm thấy trên mạng. Ví dụ, mã hexa của h_0 cho file video `mp4` tại địa chỉ

https://tinyurl.com/birthdayAttack

là

`03c08f4ee0b576fe319338139c045c89c3e8e9409633bea29442e21425006ea8`