Để giải quyết thử thách này, chúng ta cần khôi phục khóa AES bằng cách khai thác việc mã hóa RSA nhiều lần với các số mũ và modulus khác nhau. Mục tiêu là chọn các số mũ sao cho có thể kết hợp các bản mã để tái tạo lại khóa.

**Phân tích**

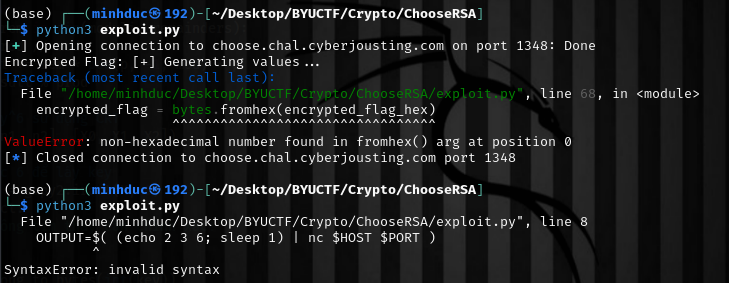
1. Cơ chế mã hóa: Khóa AES được mã hóa 3 lần bằng RSA với các modulus khác nhau (n0, n1, n2) và các số mũ e do người dùng chọn.
2. Lựa chọn số mũ: Chọn các số mũ e0=2, e1=3, e2=6 để tận dụng việc kết hợp các bản mã và sử dụng Định lý phần dư Trung Hoa (CRT) để khôi phục key^6.
3. Khai thác CRT: Tính key^6 từ các bản mã và modulus, sau đó khai căn bậc 6 để lấy khóa AES.

**Giải pháp**

1. Chọn số mũ: Chọn e0=2, e1=3, e2=6 để các bản mã có thể kết hợp thành key^6.
2. Tính toán lại key^6: Sử dụng CRT kết hợp các bản mã đã tính toán lại để có key^6.
3. Khai căn bậc 6: Tìm khóa AES bằng cách khai căn bậc 6 của kết quả từ CRT.

Giải thích

1. **Kết nối và nhận dữ liệu**: Kết nối đến server và nhận các giá trị modulus (n0, n1, n2) và các bản mã (c0, c1, c2).
2. **Tính toán key^6**: Sử dụng các số mũ đã chọn để tính toán lại key^6 từ các bản mã và kết hợp chúng bằng CRT.
3. **Khai căn và giải mã**: Khai căn bậc 6 để lấy khóa AES, sau đó giải mã flag bằng khóa này.



Lỗi xảy ra do việc đọc sai dòng chứa flag mã hoá từ server. Ta cần điều chỉnh phần đọc dữ liệu để chính xác hơn. Đây là code đã fix:

Những thay đổi chính:

1. Thêm r.recvuntil(b"[+] Generating values...\n") để bỏ qua dòng thông báo
2. Đọc chính xác dòng chứa flag hex ngay sau dòng thông báo
3. Giữ nguyên logic giải mã nhưng đảm bảo đúng thứ tự đọc dữ liệu

