Online Game Security

Part I. Network Security

목표

- 네트워크와 패킷 공격을 방어
- 공격 형태
  - 패킷 누락 (Packet Drop)
  - 패킷 삽입 (Packet Insertion)
  - 패킷 변경 (Packet Modification)
  - 패킷 도청 (Eavesdropping)
  - 패킷 생성 (Packet Generation / Replay)
- 서버 공격
  - DDoS
    - SYN Flooding
    - Teardrop Attacks
    - Application layer floods
    - …
  - 서버 해킹
    - 장비에 들어가기
    - 다양한 방법
    - Network Engineer들과 퍼블리셔의 역할

패킷 공격

- 논클라 봇
  - Non-Client Bots
  - C9 게임 플레이
  - MU 클라이언트 + 오토 플레이 + 핵 기능
- 아이템 복제
  - MU 상점 거래와 서버 이동을 패킷을 빠르게 전송하여 공격
- 패킷 변경
  - 메모리 상에서 패킷으로 만들기 전에 주로 이루어짐
  - 따라서, 메모리 조작을 포함해서 생각해야 함

패킷 보호

- 체크섬 (Checksum)
  - 복잡하면 성능 이슈가 있고 간단하면 조작할 수 있다.
  - 따라서, 체크섬에 의존하기는 어렵다.
- 암호화 (Encryption)
  - CryptoPP (C++)
    - 뮤2 안드로이드 빌드 이슈로 인해 제거되어 있는 상태
- 순서 체크 (Sequence Check)
  - Drop / Insertion 방지

암호화

- 키 형태에 따른 분류
  - 대칭 키 (Symmetric Key) 알고리즘
    - TEA, DES, 3DES 등등 매우 많음
  - 공개 키 (Public Key) 알고리즘
    - RSA / DSA
    - 키 배포에 주로 사용
      - 공인인증서
- 게임에서 암호화
  - 지키려는 사람이 깨려고 함
    - 게임의 고유한 특성
    - 키 기반 암호화를 무력화 시킴
  - 따라서, 난독화 / 동적인 변경이 필요

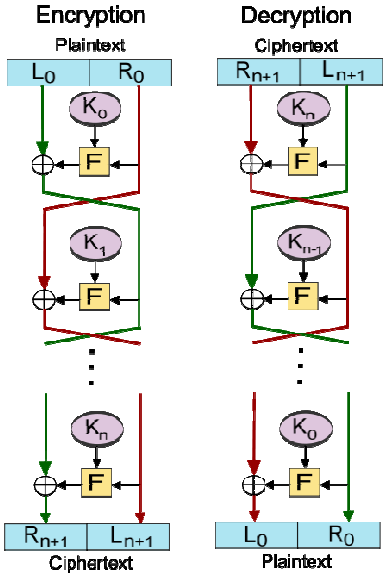
게임 패킷의 암호화

- 성능(암호화 속도와 CPU 사용량)이 중요
- 사용자 마다 다른 키 / 알고리즘 사용
- 동적으로 키 / 알고리즘을 변경
- 추가적으로 분석하기 힘들고 변경이 잦은 암호화 필요
- 선택들
  - 통신 데이터에 따라 주기적으로 키 / 알고리즘을 선택
  - 많은 알고리즘들 중 빠른 것은 오래 사용하고 느린 것은 잠깐 사용
  - 많은 패킷들 중 이동과 같이 잦은 것들은 제외

암호화

- http://math.scu.edu/~eschaefe/book.pdf
  - 기본적인 소개
- Transposition (위치) / Substitution (교체)
- TEA
  - Feistel Cipher 계열
  - 여러 번 (Round) 키 / 데이터 변환

```
void encrypt (uint32_t* v, uint32_t* k) {
    uint32_t v0=v[0], v1=v[1], sum=0, i;           /* set up */
    uint32_t delta=0x9e3779b9;                     /* a key schedule constant */
    uint32_t k0=k[0], k1=k[1], k2=k[2], k3=k[3];   /* cache key */
    for (i=0; i < 32; i++) {                       /* basic cycle start */
        sum += delta;
        v0 += ((v1<<4) + k0) ^ (v1 + sum) ^ ((v1>>5) + k1);
        v1 += ((v0<<4) + k2) ^ (v0 + sum) ^ ((v0>>5) + k3);
    }                                              /* end cycle */
    v[0]=v0; v[1]=v1;
}
```



Packet Twister

- 단순한 아이디어
  - 추가적으로 분석하기 힘들고 변경이 잦은 암호화 필요
  - 변환 단위를 Op (Operation)으로 하고
  - 이들을 섞어서 변환을 만듦
  - 패킷 타잎 별로 생성된 코드 고정
  - 그래서, 단순하다

Packet Twister

- 개선 방향
  - Op 추가
    - 좀 더 많은 Transposition과 Substitution 추가
    - Feistel Cipher의 개념 차용
      - Rotation shift
      - XOR data table
      - Rounds (적용 횟수)
  - 변경 부위 확장
    - 현재는 앞 쪽에만 적용
    - 패킷 접기
  - 동적인 변화
    - 암호화 구현의 아이디어 차용
    - 패킷 타잎 별로 동적인 변화
      - 함수 변경