Architektonické závady.

- Pro každého klienta je vygenerován sdílený klíč, kterým jsou potom chráněny licence. Server si musí udržovat velký seznam klíčů. Při kompromitaci serveru jsou automaticky kompromitováni i všichni klienti.
- Balíky se šifrují pomocí AES-CBC s nulovým paddingem. Nelze dešifrovat se správnou velikostí.
- Licence nejsou podepsané, lze je tedy falšovat (parser licencí nebyl dodán).

Implementace datového serveru.

- Klíče nejsou uloženy v X.509, ukládá se p, q, n v base16, klíč se dopočítává.
- PRNG se instaciuje před každým generováním – spotřebovává se entropie, případně mohou být key a iv korelovány.
- Data_server::sign_file_private_rsa volá rsa_private se vstupem špatné délky – out of bounds access. Součástí podpisu je kus zásobníku licenčního serveru.
- Balíky se načítají do paměti v celku – crash serveru v případě, že je balík příliš velký.
- Testy na existenci a následné použití souboru s přeotevřením (race).
- Mnohonásobné otevírání souborů.

Implementace licenčního serveru.

- Kvadruplikace funkcionality při načítání permissions.
- string::substr

```
string file_name;
for(unsigned int i=path.find_last_of('\\')+1; i<path.length(); i++)
  file_name += path.at(i);
```

Rants.

- `malloc`, `free`.
- Nulování pole.

```
std::fill(initialize_vector, initialize_vector + iv_size, 0);
```

- Místo

```
bool pom = false;
do{
  if(x){
    pom = true;
  }
  else{
    foo();
  }
} while (false == pom);
```

lze psát

```
for (;;) {
  if (x)
    break;
  foo();
}
```

- Velké objekty je třeba předávat referencí.

  ```
  void set_client_data_from_database(string clientID);
  ```

  ```
  void set_client_data_from_database(string const & clientID);
  ```

- Top-level const se ignoruje.

  ```
  void encrypt_licence_file_aes_cbc(const string path);
  ```

- ```
  char* pch = (char*)malloc( sizeof( char ) *(s.length() +1) );
  strcpy( pch, s.c_str() );
  TiXmlText * text = new TiXmlText(pch);
  ```

- ```
      //free
      free(pch);
  ```