

Testování Project + protocol

Datum testování: 29.3.2024

Testovaný commit: 50329d5c0bc7afaba45f36cf0c77645cd64d9173

Prostředí: Windows 10 Pro, Qt Creator 12.0.2, Qt 6.6.2 MinGW 64-bit

Projekty

- Spustím TraceXpert a kliknu na Close project, program spadne. (předpokládám, že se jedná o stejnou chybu jako je další bod)
- Spustím TraceXpert, vytvořím nový projekt a hned dám Close project, program spadne
 - Pokud po vytvoření projektu otevřu protocol manager, dám close project, tak program nespadne
- Replikace pádu programu:
 - otevři Project 1 (umístěn ve složce sources),
 - dvojklikni na uniform bytes pod TRandomPlugin,
 - initialize
 - problém bude patrně spojen s ukládáním projektů, protože jsem nebyl schopen chybu napodobit v rámci nového projektu
- Pod záložku View bych přidal i protocol managera

Protokoly

- Je potřeba u boolean určovat endianitu? Nevím, jaký datový typ se pro to používá. (v minulém testu jsem ho zapomnul zahrnout do výčtu typů, co je nepotřebují)
- Nelze přjmout big-endian hodnotu přes protokol. Chyba bude, že messagePart.getValue() nebene v potaz endianitu a porovnává tak špatné endianity. V proměnné na obrázku níže je dvojka v big-endian (posloupnost bytů má mít na začátku nuly):

```
tprotocol.h
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147

tryMatchResponse() {
    qWarning("Failed to fill in payload while matching message!");
    isMatch = false;
    break;
} else {
    // qInfo("Payload filled in ok...");
}
else {
    if(messagePart.getValue() != receivedData.sliced(pos, len)) {
        // qInfo("Header message part does not match...");
        isMatch = false;
        break;
    } else {
        // qInfo("Header message part match!");
    }
}
pos += len;

if(isMatch && pos != receivedData.length()) {
    // qInfo("Could be a match, but received message has yet more data...");
}
```

Name	Type	Value
> _for_range	<2 items>	
> _for_range@1	<2 items>	
iok	true	true
isMatch	true	true
len	4	4
> message	@0xe2241fb10	
> messagePart	@0x2100e5beaa0	
m_description	''	
m_hasStaticLength	true	true
m_isLittleEndian	false	false
m_isPayload	false	false
m_length	4	4
> m_name	"BE"	
m_state	TMessagePart::TState::Tok (0)	
m_stateMessage	''	
m_type	TMessagPart::TType::TInt (8)	
> m_value	\002\000\000\000\000\000\000\002	
pos	4	4
> receivedData	'\001\000\000\000\000\000\000\002'	
> *this	@0xe2241fc020	

- Jen nápad: když odesílám data proměnlivé délky, program si může sám vypočítat délku dat. Program při odeslání kontroluje, zda délka souhlasí s daty. To by rovnou mohl délku sám doplnit. Například, když posílám string, musím vyplnit délku stringu a string samotný. Přitom program v tu dobu už string zná. Z pohledu user experience mě nevádí počítat délky, které nejsou nutně potřeba. Osobně bych volil:
 - Samotnou doplněnou délku bych neskrýval v send IODevice, ale dal bych ji readonly (pouze pro GUI) s aktualizací při změně dat (daná aktualizace formuláře tam již je).
 - Udělat podmínku při kontrole příkazu (pokud to jednoduše lze): když je délka nastavena, tak ji zkontolovat, pokud není, tak ji doplnit. Tohle bude vhodné u scénáře, kde se může délka předávat (pokud ji známe). A pokud ji neznáme, pak výpočet necháme na protokolech.
- Z minulého testování:
 - Pokud protokol nemá command, v IODevice mám možnost vybrat zprávu z prázdné nabídky. Nešlo by v takovém případě vypsat „Protokol neobsahuje žádné příkazy“? At uživatel hned ví, co se děje. **Implementováno**
 - Nevidím, že by to bylo implementováno:

The screenshot shows a window titled "Send data". Inside, there are two dropdown menus: "Protocol" which is currently set to "bez cmd", and "Message" which is empty. Below these is a large, empty text input field and a "Send" button at the bottom.

TRandomPlugin

- Defaultní hodnotu seedu v pre-init parametru bych nastavil na 0.