Descrizione di come il malware ottiene la persistenza, con evidenziazione del codice assembly.
Identificazione del client software utilizzato per la connessione a Internet.
Identificazione dell'URL a cui il malware tenta di connettersi, con evidenziazione della chiamata di funzione per la connessione.
Bonus: Significato e funzionamento del comando assembly "lea".

1. Persistenza del malware e codice assembly:

```
)040286F   push    2               ; samDesired
)0402871   push    eax             ; ulOptions
)0402872   push    offset SubKey   ; "Software\\Microsoft\\Windows\\CurrentVersion\\Run"
)0402877   push    HKEY_LOCAL_MACHINE ; hKey
)040287C   call    esi ; RegOpenKeyExW
)040287E   test    eax, eax
)0402880   jnz     short loc_4028C5
)0402882
)0402882 loc_402882:
)0402882   lea     ecx, [esp+424h+Data]
)0402886   push    ecx             ; lpString
)0402887   mov     bl, 1
)0402889   call    ds:lstrlenW
)040288F   lea     edx, [eax+eax+2]
)0402893   push    edx             ; cbData
)0402894   mov     edx, [esp+428h+hKey]
)0402898   lea     eax, [esp+428h+Data]
)040289C   push    eax             ; lpData
)040289D   push    1               ; dwType
)040289F   push    0               ; Reserved
)04028A1   lea     ecx, [esp+434h+ValueName]
)04028A8   push    ecx             ; lpValueName
)04028A9   push    edx             ; hKey
)04028AA   call    ds:RegSetValueExW
```

Il malware ottiene la persistenza attraverso varie tecniche, tra cui l'aggiunta di voci di registro, la modifica dei file di avvio del sistema operativo, l'installazione di servizi o la creazione di schedulazioni di attività. Ecco un esempio di codice assembly che potrebbe essere utilizzato per modificare il registro di Windows al fine di ottenere la persistenza:

```
mov     eax, [ebp+arg_0]     ; Carica l'indirizzo del percorso del malware in eax
push    eax                  ; Push dell'indirizzo del percorso del malware
call    ds:RegSetValueExA    ;Chiama la funzione RegSetValueExA per scrivere nel registro
```

Nel codice sopra, RegSetValueExA è una funzione di Windows API utilizzata per scrivere un valore nel registro. Il malware potrebbe utilizzare questa funzione per aggiungere una voce di registro che garantisca la sua esecuzione al riavvio del sistema.

2. Client software per la connessione a Internet:

Il malware potrebbe utilizzare librerie di terze parti per la connessione a Internet, come WinINet o Winsock, o potrebbe implementare il proprio stack di rete.

```
; DWORD __stdcall StartAddress(LPVOID)
StartAddress    proc near               ; DATA XREF: sub_401040+EC↑o
                push    esi
                push    edi
                push    0               ; dwFlags
                push    0               ; lpszProxyBypass
                push    0               ; lpszProxy
                push    1               ; dwAccessType
                push    offset szAgent  ; "Internet Explorer 8.0"
                call    ds:InternetOpenA
                mov     edi, ds:InternetOpenUrlA
                mov     esi, eax
```

nel nostro caso selezionerà internet explorer 8.0 per accedere

3. URL a cui il malware tenta di connettersi e chiamata di funzione:

Per identificare l'URL a cui il malware tenta di connettersi, è necessario esaminare il codice sorgente o il codice assembly del malware stesso. Ecco un esempio di codice assembly che potrebbe essere utilizzato per connettersi a un URL:

```
lea     edx, [ebp+url]       ; Carica l'indirizzo dell'URL nella memoria
push    edx                  ; Push dell'indirizzo dell'URL
call    ds:InternetOpenUrlA  ; Chiama la funzione InternetOpenUrlA per connettersi all'URL
```

In questo esempio, InternetOpenUrlA è una funzione di Windows API utilizzata per aprire un'URL in una sessione Internet.

```
loc_40116D:                             ; CODE XREF: StartAddress+30↓j
                push    0               ; dwContext
                push    80000000h       ; dwFlags
                push    0               ; dwHeadersLength
                push    0               ; lpszHeaders
                push    offset szUrl    ; "http://www.malware12COM
                push    esi             ; hInternet
                call    edi ; InternetOpenUrlA
                jmp     short loc_40116D
StartAddress    endp
```

4. Bonus: Significato e funzionamento del comando assembly "lea":

Il comando lea (Load Effective Address) in assembly viene utilizzato per caricare l'indirizzo effettivo di una variabile o di un'area di memoria in un registro senza effettuare il dereferenziamento della variabile stessa. In altre parole, lea calcola l'indirizzo di memoria di un'area e lo carica in un registro, ma non accede effettivamente ai dati memorizzati in quell'indirizzo.