第 7 回ソフトウェア翻訳士認定試験

＜一次試験＞　6 月 3（日）10：00～15：00

問題 1・2 の両方について解答のこと。選択ではありません。

＜問題1＞　下線部を訳して提出してください。

[BLOCK-1]

SYN Attacks: Denial of Service

A *SYN attack* is an attack against a computer that provides service to customers over the Internet. *SYN* refers to the type of message (Synchronize) that is used between computers when a network connection is being made. In this attack, the enemy runs a program from a remote location (anywhere in the world) that jams the service on the victim computer. This is known as a *denial-of-service attack* because the effect of the attack is to prevent the service-providing computer from providing the service. The attack might prevent one site from being able to exchange data with other sites or prevent the site from using the Internet at all. Increasingly, companies are depending on Internet services for day-to-day business, from email to advertising to online product delivery. Some companies' business is entirely dependent on the Internet.

[BLOCK-2]

SYN attacks have been used successfully against a wide variety of targets, but they have the greatest impact against the companies that provide connections to the Internet. These Internet service providers, or ISPs, provide Internet connection services to government, businesses, and individuals. A SYN attack against an ISP usually results in disruption of Internet service to all the service provider's customers.

This type of attack is very difficult to prevent because it exploits a design flaw in the basic technology used for Internet communication today. Experts are currently working on techniques to reduce the problem somewhat, but preventing these attacks from occurring in the future will require a change in the way Internet communications are accomplished by the computers using the Internet. This is likely to take several years.

[BLOCK-3]

IP Spoofing: Masquerading

In an attack known as *IP spoofing,* attackers run a software tool that creates Internet messages that appear to come, not from the intruder's actual location, but from a computer trusted by the victim. *IP,* which stands for Internet Protocol, refers to the unique address of a computer. When two computers trust each other, they allow access to sensitive information that is not generally available to other computer systems. The attacker takes advantage of this trust by masquerading as the trusted computer to gain access to sensitive areas or take control of the victim computer by running "privileged" programs. Information that has been

compromised through IP spoofing includes credit card information from a major Internet service provider and exploitation scripts that a legitimate user had on hand for a security analysis.
Unfortunately, there are many computer programs and services that rely on other computers to “speak the truth” about their address and have no other mechanism for disallowing access to sensitive information and programs. The CERT Coordination Center has received many reports of attacks in which intruders (even novice intruders) used this technique to gain access to computer systems with the help of publicly available IP spoofing computer programs.

This attack technique is being addressed by fundamental changes in the way computers communicate over the Internet. The IETF (Internet Engineering Task Force) Proposed Standard for the Next Generation Internet Protocol (IPng) is being designed to provide integral support for authenticating hosts and protecting the integrity and confidentiality of data.

Although early implementations of IPng are underway, the IP spoofing technique is likely to remain effective for years.

＜問題2＞　下線部を訳して提出してください。

Asynchronous Database Access with Qt 4.x
How to code around the default synchronous database access in Qt4.

The database support in Qt 4.x is quite robust. The library includes drivers for Oracle, PostgreSQL, SQLite and many other relational databases. Out of the box, the Qt database library also contains bindings for many widgets and provides data types for the transparent handling of result sets coming from a database. But, your application can pay a price for these conveniences. All database access is synchronous by default, which means that intensive and time-consuming SQL queries normally will lock up the UI unless precautions are taken. Using stored procedures on the server can sometimes help the situation; however, this is not always possible or desirable. And often, the length and cost of the queries generated by your application simply cannot be known in advance, so the door is left open for undesirable UI behavior. People don't want their application to "lock up" at odd moments; however, this is the default behavior, and so we must contend with it.
Fortunately, Qt 4.x also has robust support for multithreaded programming. By placing the heavy-duty database work in separate threads, the UI is free to respond to the user normally, without ungraceful interruptions. As with all concurrent programming, however, you must take precautions to ensure the correct sequence of interactions between threads. For example, when sharing data among threads, guard it properly using mutexes. When communicating between threads, consider carefully how the interaction will behave, and in what sequence. In addition, when utilizing a database connection within a thread separate from the UI thread, you must pay attention to some extra caveats. A proper implementation that keeps certain things in mind will make significant improvements in the UI behavior and responsiveness of a database application.

Thread Strategies
There are several ways to distribute the database load to separate threads of execution. Fortunately, all of them share the same characteristics when it comes to the details of creating and using a database connection properly. The primary consideration is to use a database connection only within the thread that created it. For regular synchronous applications, the default behavior is fine. The QSqlDatabase::addDatabase() static function creates a database connection within the context of the application's main UI thread. Queries executed within this same thread will then cause blocking behavior. This is to be expected.

In order to run queries in parallel with the main UI thread, so that they do not interrupt the main event processing loop, a database connection must be established in the thread in which the query execution, which should be separate from the main UI thread. However you structure the threading in your application, your design must be able to establish a connection within the context of each thread that will be performing database work.

For example, creating a thread pool in which a few threads handle the road of querying the database in a round-robin fashion (without the overhead of creating and destroying threads all the time) will push the time-consuming work outside the main event loop. Or, depending on the needs of your application, you simply can spawn threads on an as-needed basis to perform database work. In either case, you must create a connection per thread.

以上