```cpp
#include <iostream>
#include <string>
#include <vector>
#include <ctime>
#include <cstdlib>
#include <algorithm>

bool hasUpperCase(const std::string& password) {
    return std::any_of(password.begin(), password.end(), ::isupper);
}

bool hasLowerCase(const std::string& password) {
    return std::any_of(password.begin(), password.end(), ::islower);
}

bool hasDigit(const std::string& password) {
    return std::any_of(password.begin(), password.end(), ::isdigit);
}

bool hasSpecialChar(const std::string& password) {
    return password.find_first_of("!@#$%^&*()") != std::string::npos;
}

std::string generateStrongPassword(const std::string& input) {
    std::string generated = input;

    // Add missing character types
    if (!hasUpperCase(generated)) {
        generated += 'A' + (rand() % 26); // Add a random uppercase letter
    }
    if (!hasLowerCase(generated)) {
        generated += 'a' + (rand() % 26); // Add a random lowercase letter
    }
    if (!hasDigit(generated)) {
        generated += '0' + (rand() % 10); // Add a random digit
    }
    if (!hasSpecialChar(generated)) {
        generated += "!@#$%^&*()"[rand() % 10]; // Add a random special character
    }

    // Shuffle the generated password to ensure randomness
    std::random_shuffle(generated.begin(), generated.end());

    // Ensure minimum length
    while (generated.length() < 12) {
        generated += 'A' + (rand() % 26); // Add random characters to meet length
    }

    return generated;
}

int main() {
    srand(static_cast<unsigned int>(time(0))); // Seed for randomness
    std::string password;

    std::cout << "Enter a password to improve its strength: ";
    std::cin >> password;

    std::string strongPassword = generateStrongPassword(password);
    std::cout << "Suggested stronger password: " << strongPassword << std::endl;

    return 0;
}

```