Secure Coding

STRING

- 1 被污染的数据
- 2 动态分配字符串
- 3 字符串对象引用失效

1 被污染的数据

如果一个值被称为被污染的，如果它的来源是不受信任的（程序的控制之外），并且没有被净化，以确保它符合该值的使用者要求的约束。

例如，所有字符串都要求是空字符结尾的约束。

```
#include <windows.h>
#include <iostream>
using namespace std;
bool IsPasswordOK(void)
{
	char Password[12];
	gets_s(Password);
	return 0==strcmp(Password,"goodpass");
}
int main( )
{

	bool PwStatus;
	puts("Enter password:");
	PwStatus=IsPasswordOK();
	if(PwStatus==false)
	{
		puts("Access denied");
	}
	system("pause");
	return 0;
}
```

1 被污染的数据

缺陷1

char Password[12] 只能容纳11个字符

缺陷2

没有检查gets_s(Password) 的返回状态

当gets()失败是，Password缓冲区的内容是不确定的。

1 被污染的数据

缺陷3

存在绕过缺陷。

```
#include <windows.h>
#include <iostream>
using namespace std;
int main( )
{
	bool PwStatus;
	char Password[12];
	puts("Enter password:");
	cin>>Password;//此处输入任意字符
	PwStatus=!strcmp(Password,"goodpass");
	//cin>>Password;//此处输入24个1
	if(PwStatus==false)
	{
		puts("Access denied");
	}
	system("pause");
	return 0;
}
```

2 动态分配字符串

使用new动态分配字符串内存，要确保内存操作不越界。

```
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <iostream>

using namespace std;

int main( )
{
	char *p=new char[4];

	cin.getline(p,20);
	cout<<p<<endl;
	system("pause");
	return 0;
}
```

3 字符串对象引用失效

修改字符串的操作会使引用、指针和引用字符串对象的迭代器失效。

```
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
	char input[10];
	string email;
	string::iterator loc=email.begin();
	cin>>input;
	for(size_t i=0;i<strlen(input);i++)
	{
		if(input[i]!=';')
				email.insert(loc++,input[i]);
		else
				email.insert(loc++,' ');
	}
	cout<<email<<endl;
	system("pause");
	return 0;
}
```

```
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
	char input[10];
	string email;
	string::iterator loc=email.begin();
	cin>>input;
	for(size_t i=0;i<strlen(input);i++)
	{
		if(input[i]!=';')
			email.insert(loc++,input[i]);
		else
			email.insert(loc++,' ');
	}
	cout<<email<<endl;
	system("pause");
	return 0;
}
```

```
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
	char input[10];
	string email;
	string::iterator loc=email.begin();
	cin>>input;
	for(size_t i=0;i<strlen(input);i++)
	{
		if(input[i]!=';')
			email.insert(loc,input[i]);
		else
			email.insert(loc,' ');
		++loc;
	}
	cout<<email<<endl;
	system("pause");
	return 0;
}
```