人怜直节生来瘦，自许高材老更刚。

专注web、移动应用安全研究。————求Android安全相关工作！

昵称：bamb00
园龄：3年10个月
粉丝：49
关注：1
+加关注

< 2011年7月 >

日	一	二	三	四	五	六
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

搜索

找找看

谷歌搜索

常用链接

C++类型转换总结

C风格的强制类型转换(Type Cast)很简单，不管什么类型的转换统统是：
TYPE b = (TYPE)a。
C++风格的类型转换提供了4种类型转换操作符来应对不同场合的应用。

const_cast，字面上理解就是去const属性。
static_cast，命名上理解是静态类型转换。如int转换成char。
dynamic_cast，命名上理解是动态类型转换。如子类和父类之间的多态类型转换。
reinterpret_cast，仅仅重新解释类型，但没有进行二进制的转换。
4种类型转换的格式，如：TYPE B = static_cast(TYPE)(a)。

const_cast

去掉类型的const或volatile属性。

```
1 struct SA {
2 int i;
3 };
4 const SA ra;
5 //ra.i = 10; //直接修改const类型，编译错误
6 SA &rb = const_cast<SA&>(ra);
7 rb.i =10;
```

我的随笔
我的评论
我的参与
最新评论
我的标签
更多链接

我的标签

python(16)
uliweb(5)
django(3)
xss(3)
linux(3)
excel(2)
漏洞(2)
面试题(2)

static_cast

类似于C风格的强制转换。无条件转换，静态类型转换。用于：
1. 基类和子类之间转换：其中子类指针转换成父类指针是安全的；但父类指针转换成子类指针是不安全的。(基类和子类之间的动态类型转换建议用dynamic_cast)
2. 基本数据类型转换。enum, struct, int, char, float等。static_cast不能进行无关类型（如非基类和子类）指针之间的转换。
3. 把空指针转换成目标类型的空指针。
4. 把任何类型的表达式转换成void类型。
5. static_cast不能去掉类型的const、volitale属性(用const_cast)。

```
int n =6;
double d = static_cast<double>(n); // 基本类型转换
int*pn =&n;
double*d = static_cast<double*>(&n) //无关类型指针转换，编译错误
void*p = static_cast<void*>(pn); //任意类型转换成void类型
```

dynamic_cast

有条件转换，动态类型转换，运行时类型安全检查(转换失败返回NULL)：
1. 安全的基类和子类之间转换。
2. 必须要有虚函数。
3. 相同基类不同子类之间的交叉转换。但结果是NULL。

```
class BaseClass {
public:
int m_iNum;
virtualvoid foo(){}; //基类必须有虚函数。保持多台特性才能使用dynamic_cast
};

class DerivedClass: public BaseClass {
public:
char*m_szName[100];
void bar(){};
```

url(2)

URL提取(1)

更多

随笔分类

android安全(29)

Android底层(5)

android开发(4)

C/C++(3)

django(4)

Linux(5)

python(40)

WEB安全(13)

数据结构与算法(4)

随笔档案

2015年1月 (3)

```
11 };
12
13 BaseClass* pb =new DerivedClass();
14 DerivedClass *pd1 = static_cast<DerivedClass *>(pb); //子类->父类，静态类型转换，正确但不推荐
15 DerivedClass *pd2 = dynamic_cast<DerivedClass *>(pb); //子类->父类，动态类型转换，正确
16
17 BaseClass* pb2 =new BaseClass();
18 DerivedClass *pd21 = static_cast<DerivedClass *>(pb2); //父类->子类，静态类型转换，危险！访问子类m_szName成员越界
19 DerivedClass *pd22 = dynamic_cast<DerivedClass *>(pb2); //父类->子类，动态类型转换，安全的。结果是NULL
```

reinterpret_cast

仅仅重新解释类型，但没有进行二进制的转换：

1. 转换的类型必须是一个指针、引用、算术类型、函数指针或者成员指针。
2. 在比特位级别上进行转换。它可以把一个指针转换成一个整数，也可以把一个整数转换成一个指针（先把一个指针转换成一个整数，在把该整数转换成原类型的指针，还可以得到原先的指针值）。但不能将非32bit的实例转成指针。
3. 最普通的用途就是在函数指针类型之间进行转换。
4. 很难保证移植性。

```
1 int doSomething(){return0;};
2 typedef void(*FuncPtr)(); //FuncPtr is 一个指向函数的指针，该函数没有参数，返回值类型为 void
3 FuncPtr funcPtrArray[10]; //10个FuncPtrs指针的数组 让我们假设你希望（因为某些莫名其妙的原因）把一个指向下面函数的指针存入funcPtrArray数组:
4
5 funcPtrArray[0] =&doSomething;// 编译错误！类型不匹配，reinterpret_cast可以让编译器以你的方法去看待它们: funcPtrArray
6 funcPtrArray[0] = reinterpret_cast<FuncPtr>(&doSomething); //不同函数指针类型之间进行转换
```

总结

去const属性用const_cast。

基本类型转换用static_cast。

2014年11月 (2)
2014年10月 (6)
2014年9月 (8)
2014年8月 (3)
2014年7月 (4)
2014年3月 (1)
2014年2月 (2)
2014年1月 (1)
2013年12月 (1)
2013年11月 (5)
2013年10月 (15)
2013年9月 (5)
2013年8月 (12)
2013年7月 (18)
2013年6月 (8)

多态类之间的类型转换用daynamic_cast。
不同类型的指针类型转换用reinterpret_cast。

分类: C/C++

bamb00
关注 - 1
粉丝 - 49
+加关注

« 上一篇：Aho-Corasick算法学习（转载）
» 下一篇：VC6.0编译驱动程序的工程设置

posted @ 2011-07-20 18:38 bamb00 阅读(24404) 评论(0) 编辑 收藏

2013年5月 (5)
2013年4月 (7)
2013年3月 (1)
2011年12月 (2)
2011年10月 (1)
2011年8月 (1)
2011年7月 (1)
2011年6月 (3)
2011年5月 (1)
2011年4月 (4)
2011年3月 (3)

最新IT新闻:
· 思科3成经理层人员被换掉了
· 这就是英国第一辆无人驾驶汽车
· 特斯拉CEO马斯克：在中国充电难是一种假象
· 螺旋蜗壳激发科学家灵感：锂电池有望迎来更持久的续航
· 阿里系网络公益捐款达2.34亿 剁手党变身活雷锋
» 更多新闻...

最新知识库文章:
· 使用2-3法则设计分布式数据访问层
· 数据清洗经验
· 设计中的变与不变
· 通俗解释「为什么数据库难以拓展」
· 手机淘宝高质量持续交付探索之路
» 更多知识库文章...

友情链接
0nly3nd's blog
9Hao's blog
c32's blog
sunysen
最新评论

1. Re:利用IDA6.6进行apk dex代码动态调试

好流弊的感觉...

--码魂

2. Re:利用备份技术获取apk本地存储数据

不明觉厉

用来破解某些app么？

--longware

3. Re:“逃离大厦”游戏的破解

怎么说也是图文的，为什么少有人来瞄个，难道都成神了，孤独啊！

--飘飘兔

4. Re:boot.img的修改

刚好五个字

--不要呵呵

5. Re:boot.img的修改

mark

--雪雨潇潇

阅读排行榜

1. C++类型转换总结(24404)

2. c++ 函数返回引用(6247)

3. python面试题大全（一）(3909)

4. apk破解心得(2927)

5. 对一个伪装成微信的加固病毒的分析(2612)

评论排行榜

1. 对一个伪装成微信的加固病毒的分析(7)

2. 反编译apk插入广告（一）(4)

3. boot.img的修改(4)

4. “逃离大厦”游戏的破解(4)

5. ptrace注入型病毒“聊天剽窃手”分析(2)

推荐排行榜

1. 对一个伪装成微信的加固病毒的分析(11)
2. C++类型转换总结(7)
3. boot.img的修改(5)
4. c++ 函数返回引用(3)
5. PySide图形界面开发（一）(3)