Subject: Web/TServ [BUG][FIXED]
Posted by hojtsy on Sun, 02 Apr 2006 10:29:41 GMT
View Forum Message <> Reply to Message

This code in Web/TServ seems to be a failed attempt at being very tricky.

```
if(*_command == '\"') {
 while(*++_command && *_command != '\"' || *++_command == '\"')
  exec.Cat(*_command);
}
```

This could crash on a certain kind of unexpected input. If the string starts with a quote, but does not contain the closing pair, the memory after the \0 is read and compared to the quote char. This could be segfault in itself, but if it accidentally equals quote, the memory is read further. I suppose that the other side of the app does not send such invalid string, but it would still be more elegant to not crash on any kind of input.

Subject: Re: bug in Web/TServ
Posted by rylek on Tue, 04 Apr 2006 20:29:57 GMT
View Forum Message <> Reply to Message

Sorry for that, the above lines should read:

```
while(*++_command && (*_command != '\"' || *++_command == '\"'))
 exec.Cat(*_command);
```

Do you think there's any hope this modified version works, at least provided the string is null-terminated? I've always thought so, but you never know...

Regards

Tomas

Subject: Re: bug in Web/TServ
Posted by hojtsy on Tue, 04 Apr 2006 21:37:53 GMT
View Forum Message <> Reply to Message

I can not find any errors in that one. But it is still very tricky I think there are some practical problems with writing tricky code like this.

problem 1: by looking at the code it is quite much non-obvious if the intention was to replace two double quotes with one double quote, or it is just an unintentional side effect/bug. The code lacks expresiveness to human readers.

problem 2: calling a variable "exec" is nasty. exec is a function in the C standard library, and even

tough it is valid to create a variable with the same name, it bewilders readers, and requires much more attention from them. I think this could be easily avoided.

Subject: Re: bug in Web/TServ
Posted by rylek on Wed, 05 Apr 2006 05:57:54 GMT
View Forum Message <> Reply to Message

I guess you're right. The bug you have found by itself suggests such code is error-prone. I'm not a big fan of three-line boolean expressions but from time to time they're really hard to resist . As concerns the 'exec' variable, you are right again. But please also note there are literally tens of thousands global functions in both Windows and Linux development environment, in API, in the various plugins and in our own code as well, so the requirement, when taken absolutely, is much easier said than done. I'm currently quite content if I manage to avoid accidentally using macros for variable names, which doesn't work at all (not mentioning half of our methods which are invisibly appended the 'A' prefix in Windows because of hosts of the Unicode-compatibility macros), or having two nested for-loops in the same control variable,

```
for(int i;;)
    for(int i;;)
        ...
```

and even this can be sometimes tricky to avoid in a 300-line routine. Returning to the programming tutorials, one should not write 300-line routines, but again I've already met a few occasions (not many, to be honest) where breaking a long function into a multitude of smaller ones seemed to me both to decrease code legibility and to make the code less efficient. Sometimes the practical work calls for a compromise and here's the one I'm offering: next time I'm doing something with TServ, if I stumble over the above discussed code, I'll rewrite the while loop and rename the 'exec' variable.

Regards

Tomas

P.S. A little final point: the aforementioned while loop is indeed tricky, but I see it as a rather elegant way to skip quote-escaped quotes in strings. Honestly I use it rather often, thankfully most of the times without the omitted parentheses. Please just try yourself to rewrite it without the complex condition and I bet you'll find the resulting code rather ugly too .
