SysSec - 10K students (10Kstudents.eu)
Notes about the code examples in the slides (or: how to do this yourself)

The code examples in the slides were all compiled in a way that makes the resulting assembly a s easy to understand as possible. When you try this on real code, you will find that things ar e a little (but perhaps not that much) more complicated. We will quickly look at some differen ces you may encounter and what you can do to remove them (using Linux and gcc as examples).

1. Stack alignment

By default gcc aligns the stack to 16 bytes (see http://stackoverflow.com/questions/672461/wha t-is-stack-alignment). As a result, you may see more space allocated on the stack than you per haps would expect. This does not necessarily make the attack more complicated, but you do need to take the extra space into account. However, most compilers allow you to modify the alignme nt. For instance, gcc allows you to specify the alignment by means of the "-mpreferred-stack-b oundary=n" option, which will try to keep the stack aligned to 2^n.

2. Variable reordering

Another complication, is that variables are not necessarily placed on the stack in the order i n which they are declared. The compiler is free to move them around, for instance for alignmen t reasons. This complicates things and you may have to probe a little harder.

3. Absence of function prologues and epilogues

The whole bit about storing and restoring the frame pointer is not always there, as the compil er can decide to not use a framepointer at all, if a function does not strictly need it. For i nstance, you may not need the frame pointer to access the local variables if you can also acce ss them relative to the stack pointer. In that case omitting the frame pointer gives you anoth er general purpose register. To instruct gcc to do this, you can use the -fomit-frame-pointer option. Likewise, you can tell gcc you really like frame pointers by specifying: -fno-omit-fr ame-pointer option.

4. Canaries

Nowadays, gcc turns on stack protection by default. Doing so may introduce canaries on the sta ck. If you do not like canaries, specify: -fno-stack-protector

5. Address space randomisation

Many modern systems randomise the stack. As a result, it becomes much more complicated to find a good address to use for the return address. This can be a real hassle. It is not an option, but you can turn off randomisation for a Linux bash shell as follows: setarch $(uname -m) -RL bash