Aanwezigheid opdracht 8 + PE contactmomenten

Deadline 8 Deadline PE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 24/04/2020 | 29/04/2020 | 1/05/2020 | 6/05/2020 | 8/05/2020 | 13/05/2020 |
| Evi |  |  |  |  |  |  |
| Joy |  |  |  |  |  |  |
| Bernd |  |  |  |  |  |  |
| Albert | ................... |  |  |  |  |  |

Aanwezig = X

Afwezig = X

Planning per contactmoment

* ~~24/04/2020: Deelopdracht 1 + 2~~
* ~~29/04/2020: Deelopdracht 3 + 4~~
* ~~1/05/2020: Remote Code Execution (PE)~~
* 6/05/2020: extra DO7
* ~~8/05/2020: RSA (PE)~~
* 13/02020: Sniffing (PE)

**Buffer overflow**  
A [buffer overflow vulnerability](https://www.solarwindsmsp.com/blog/buffer-overflow-vulnerabilities-protection) refers to memory corruption that can enable malicious RCE. Buffers are sequential memory partitions that can only contain a set amount of data. When programs do not include bounds-checking measures, input can exceed the allocated memory. When a buffer overflows, it overwrites the memory in adjacent buffers. Deploying buffer overflows intentionally can destroy useful data, cause network crashes, or replace memory with arbitrary code that the instruction pointer (manipulated through another vulnerability) later executes. [The first buffer overflow attack took place in 1988](https://www.fbi.gov/news/stories/morris-worm-30-years-since-first-major-attack-on-internet-110218). The Morris Worm, a now infamous program that was responsible for the attack, caused chaos throughout the U.S. computer system, crashing 6,000 of 60,000 computers on the internet.

**Deserialization**  
A deserialization vulnerability refers to the translation of data into a live object which can accidentally result in the execution of ordinary code. An object must be serialized—converted into binary—to travel over a network. To utilize this data, a program must deserialize it, converting it back into an object. By manipulating serialized data, unauthorized attackers can produce altered objects that cause the program to execute unintended programming upon deserialization.

**Type confusion**  
Type confusion refers to a vulnerability in which code passes off an object without checking its type. This can lead to a mismatch between the object and the type it ought to be. Attackers can take advantage of this vulnerability by writing into the memory block for an object with one type pointer and reading it with another type pointer, allowing them to execute arbitrary code.