PA193 - Secure coding principles and practices

Protecting integrity of modules and external components - LABS

Petr Švenda svenda@fi.muni.cz

CROCS
Centre for Research on Cryptography and Security

Scenario

- Your application is running on desktop PC (Windows/Linux) processing large data sets (e.g., editing privacy-sensitive video) and is:
  - Loading sensitive from configuration file (e.g., keys)
  - Storing intermediate data into temporary files
- Goal:
  - Design your own process to protect confidentiality and verify integrity of configuration files
  - Design protection of temporary files (sudden crash of application, tmp file forged by an attacker…)

Original code for refactoring

```
int main() {
  FILE * pTmpFile;
  // Open temporary files
  tmpfile_s(&pTmpFile);

  char buffer[100] = "Test";
  for (size_t i = 0; i < 50; i++) {
       fwrite("Test", strlen("Test"), sizeof(char), pTmpFile);
  }

  rewind(pTmpFile);
  fseek(pTmpFile, 30, SEEK_SET);

  memset(buffer, 0, sizeof(buffer));
  fread(buffer, sizeof(char), sizeof(buffer) - 1, pTmpFile);
  printf("%s", buffer);

  fclose(pTmpFile);

  // Remove still opened tmp files (only these opened by tmpfile / tmpfile_s)
  _rmtmp();

  return 0;
}
```