개발자들이 자주 하는 보안 실수

HACKINGCAMP 28

발표자 안수현 채서연

안수현 (movptr)

Security Researcher

한세사이버보안고등학교 네트워크보안과

KITRI BoB 11기 WhiteHat 10 선정

2023년 화이트햇 콘테스트 청소년부 2위

채서연 (che-so)

Back-end Developer

명지대학교 융합소프트웨어학부 응용소프트웨어 전공

명지대학교 인문캠퍼스 멋쟁이사자처럼 11기 대표

명지대학교 SW 경진대회 우수상 수상

취약점을 찾는 이유

버그 바운티 참가, 취약점 제보 (White Hat)

ZDI, KISA, SSD, PatchDay, 자체 바운티 등

취약점과 익스플로잇을 판매 (Gray & Black Hat)

정보기관/군, 구매 업체, 스파이웨어 업체, 블랙마켓 등

보안 컨설팅, 인하우스 보안

취약점 진단, 모의해킹, 레드팀 등

취약점 찾기

무작정 인터넷에 있는 퍼저를 그대로 이용해 퍼징한다.

소스 코드나 디컴파일러를 열고 **strcpy** 등의 함수를 검색한다.

게시글에 <script>alert("XSS")</script>를 입력한다.

취약점을 찾을 수 있을까?

취약점 찾기

무작정 인터넷에 있는 퍼저를 그대로 이용해 퍼징한다.

소스 코드나 디컴파일러를 열고 **strcpy** 등의 함수를 검색한다.

게시글에 <script>alert("XSS")</script>를 입력한다.

취약점을 찾을 수 있을까?

보안에 신경을 쓴 적이 없는 프로그램이라면 가능하다.

취약점 찾기

무작정 인터넷에 있는 퍼저를 그대로 이용해 퍼징한다.

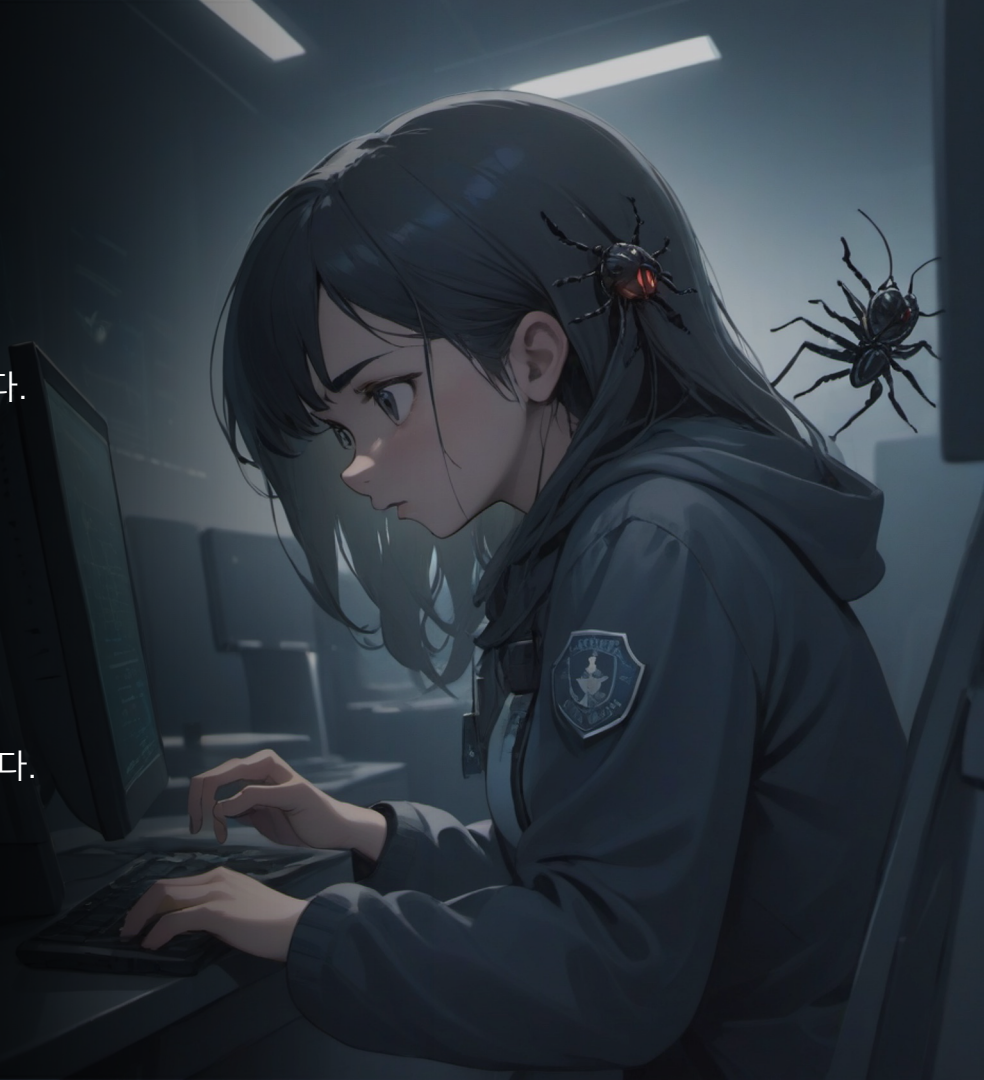
소스 코드나 디컴파일러를 열고 **strcpy** 등의 함수를 검색한다.

게시글에 <script>alert("XSS")</script>를 입력한다.

취약점을 찾을 수 있을까?

보안에 신경을 쓴 적이 없는 프로그램이라면 가능하다.

하지만, 보안에 신경을 쓴 프로그램에서도 취약점을 찾고 싶다.



취약점을 누가 만들까

취약점을 만드는 것은 당연히 **개발자**

(하지만 대부분의 경우 고의는 아니다)

그러면 개발자는 왜, 어떤 **취약점**을 만드는 것일까?

누가 이런 실수를 하겠어

Use After Free (CWE-416)

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

int main(int argc, char* argv[]) {
    uint64_t* ptr;

    ptr = (uint64_t*) calloc(256, sizeof(uint64_t));

    free(ptr);

    ptr[0] = 1;

    return 0;
}
```

누가 이런 실수를 하겠어

Use After Free (CWE-416)

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

int main(int argc, char* argv[]) {
    uint64_t* ptr;

    ptr = (uint64_t*) calloc(256, sizeof(uint64_t));

    free(ptr);

    ptr[0] = 1;

    return 0;
}
```

누가 이런 실수를 하겠어

Safe? Use After Free?

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

int main(int argc, char* argv[]) {
    uint64_t* ptr;

    ptr = (uint64_t*) calloc(256, sizeof(uint64_t));

    realloc(ptr, sizeof(uint64_t) * 1024);

    ptr[0] = 1;

    return 0;
}
```

누가 이런 실수를 하겠어

Safe? Use After Free

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

int main(int argc, char* argv[]) {
    uint64_t* ptr;

    ptr = (uint64_t*) calloc(256, sizeof(uint64_t));

    realloc(ptr, sizeof(uint64_t) * 1024);

    ptr[0] = 1;

    return 0;
}
```

누가 이런 실수를 하겠어

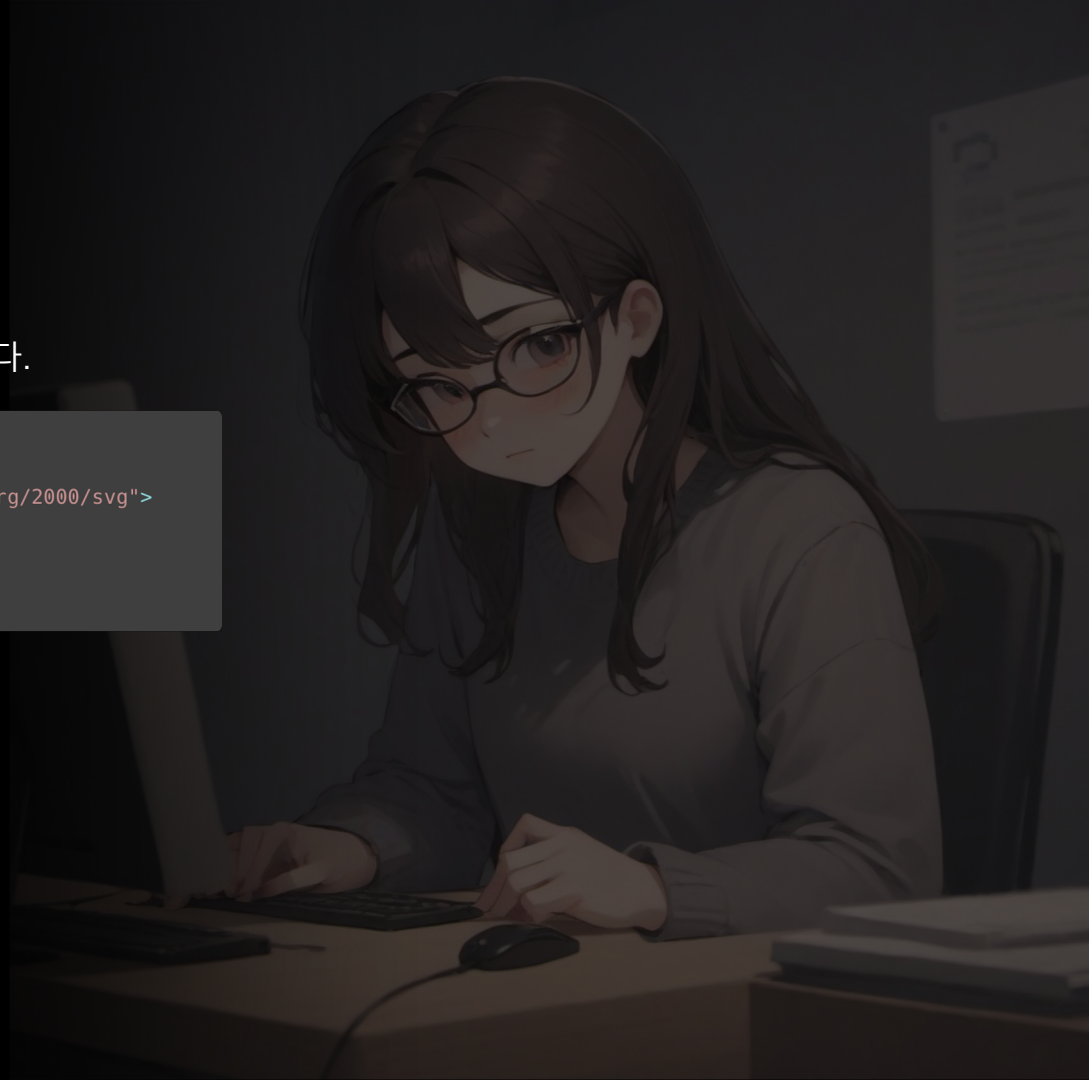
버그 클래스 버그의 유형/종류

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

int main(int argc, char* argv[]) {
    uint64_t* ptr;

    ptr = (uint64_t*) calloc(256, sizeof(uint64_t));

    realloc(ptr, sizeof(uint64_t) * 1024);

    ptr[0] = 1;

    return 0;
}
```

이게 이렇게 된다고?

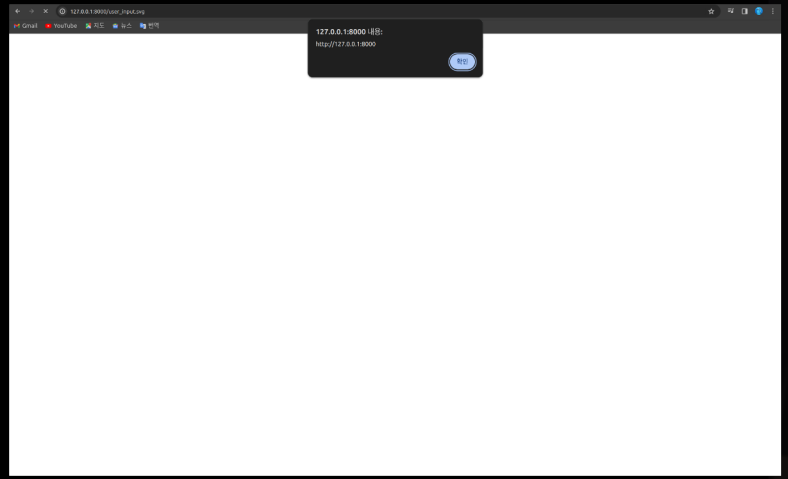
SVG 파일에서도 자바스크립트 코드를 실행할 수 있다.

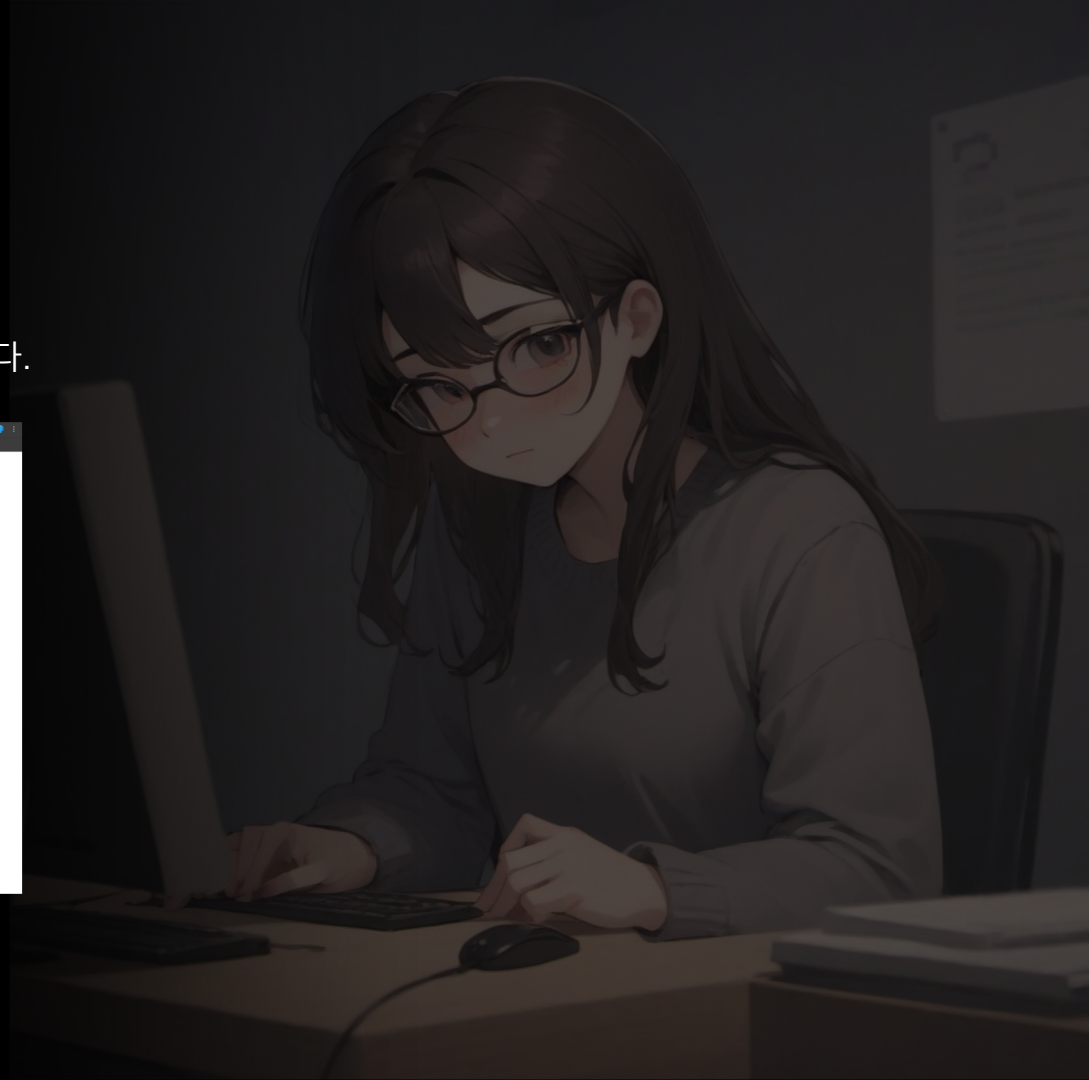
```svg
<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg">
    <script type="text/javascript">
      alert(window.origin);
    </script>
</svg>
```



이게 이렇게 된다고?

SVG 파일에서도 자바스크립트 코드를 실행할 수 있다.



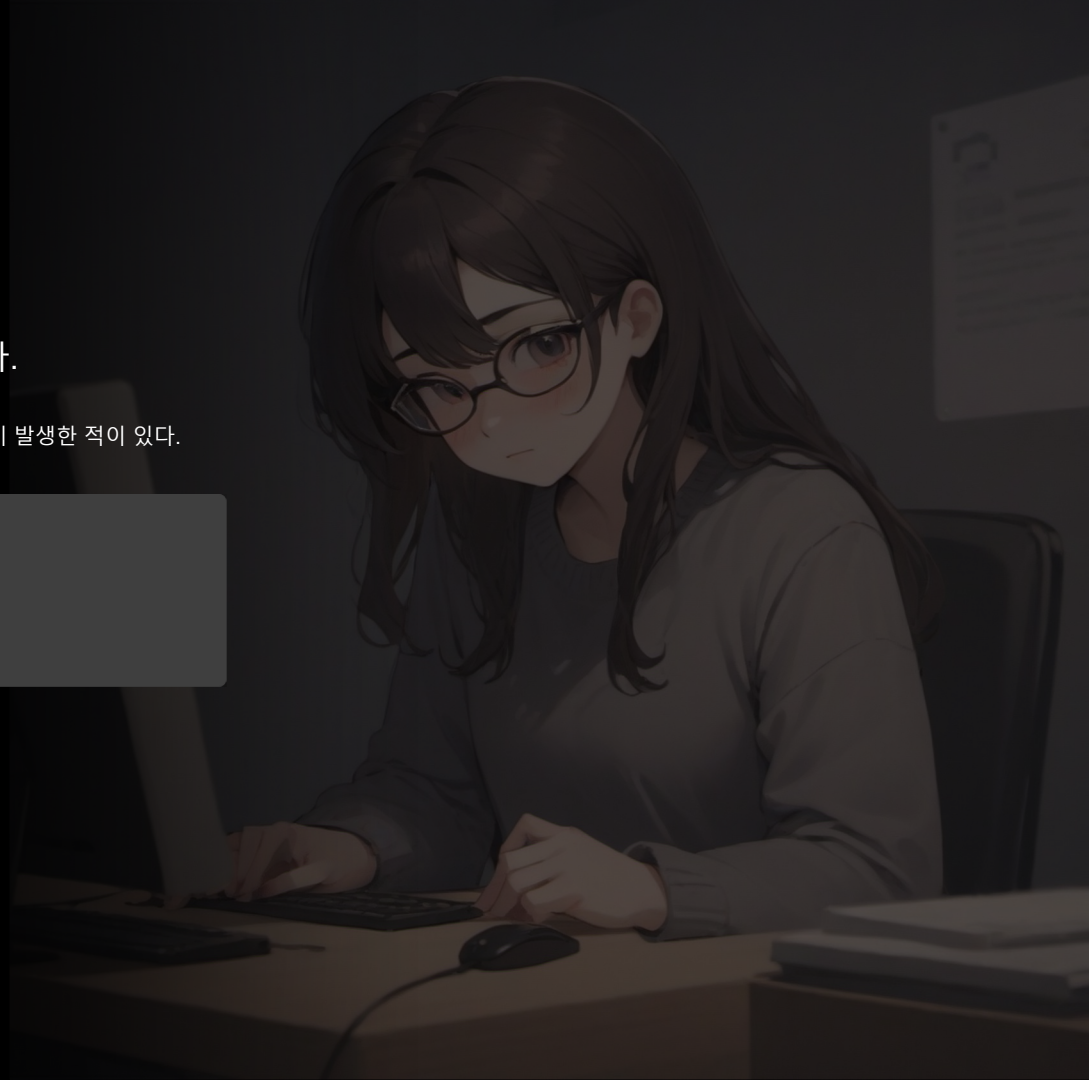


이게 이렇게 된다고?

자바스크립트 등 일부 언어에서는 0과 -0이 따로 있다.

크롬의 V8 엔진에서는 Math.expm1을 최적화할때 이를 고려하지 못해 취약점이 발생한 적이 있다.

```
Object.is(-0, -0) // true
Object.is(-0, 0) // false

-0 === 0 // true
```



이게 이렇게 된다고?

개발자가 모든 경우를 고려해서 코드를 짜기는 힘들다.

Undefined Behavior

의도와 다른 사용

오래되고 복잡한 기능

Edge Case

악용할 수 있는 기능

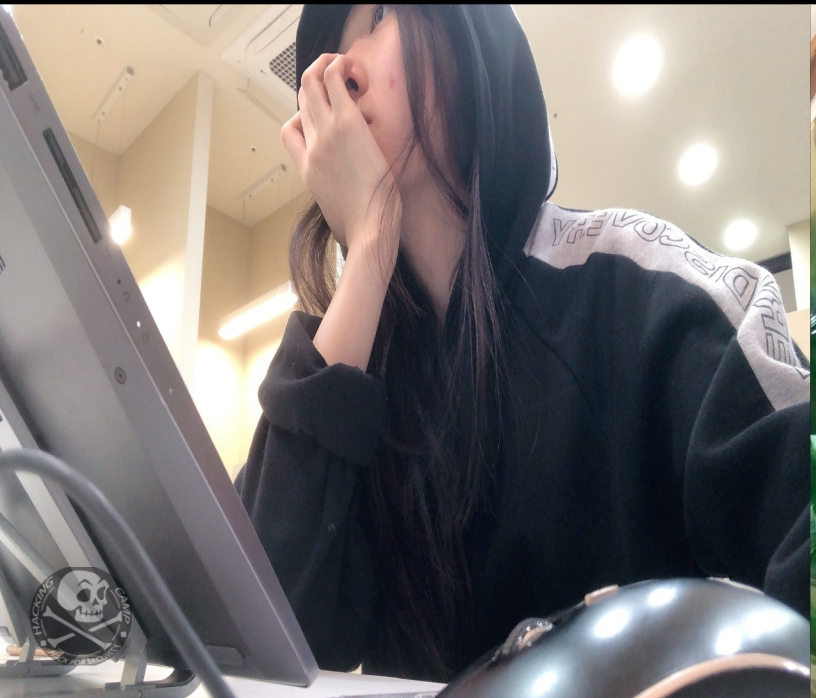
Race Condition

호환성 문제

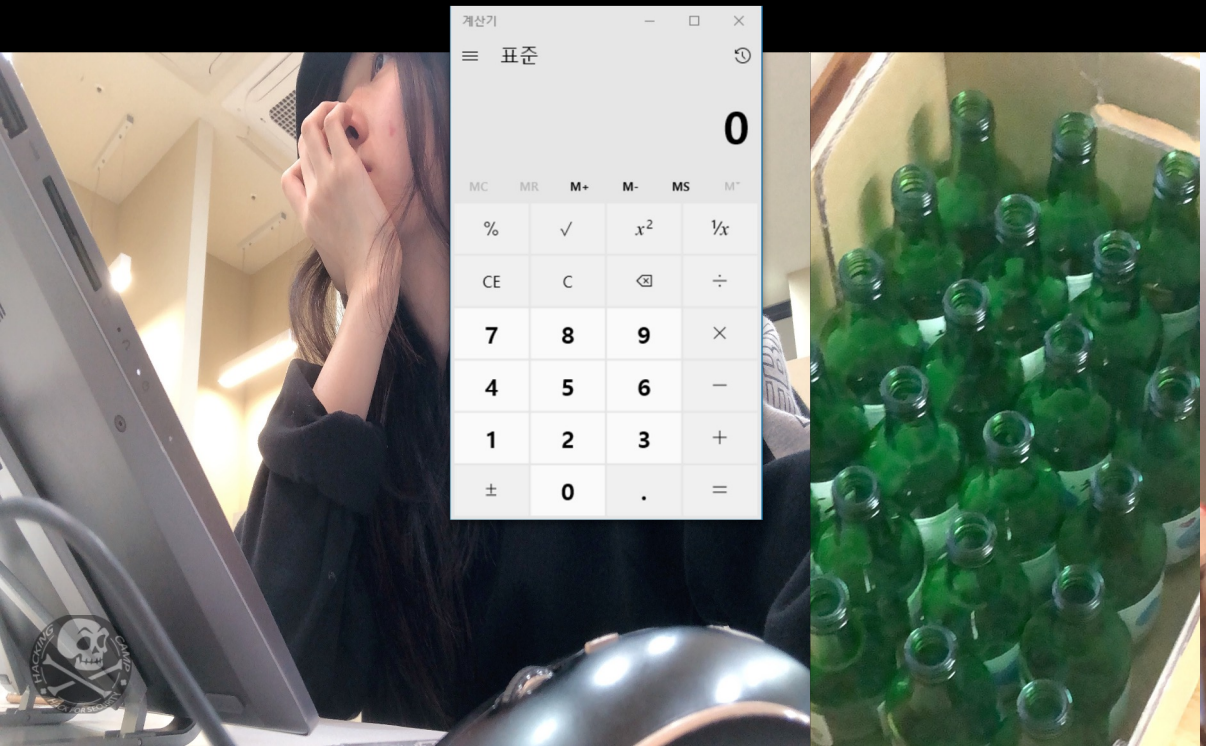
조금만 실수해도 버그나 취약점으로 이어질 수 있다.

그럼 어떻게 코드를 짜는걸까?

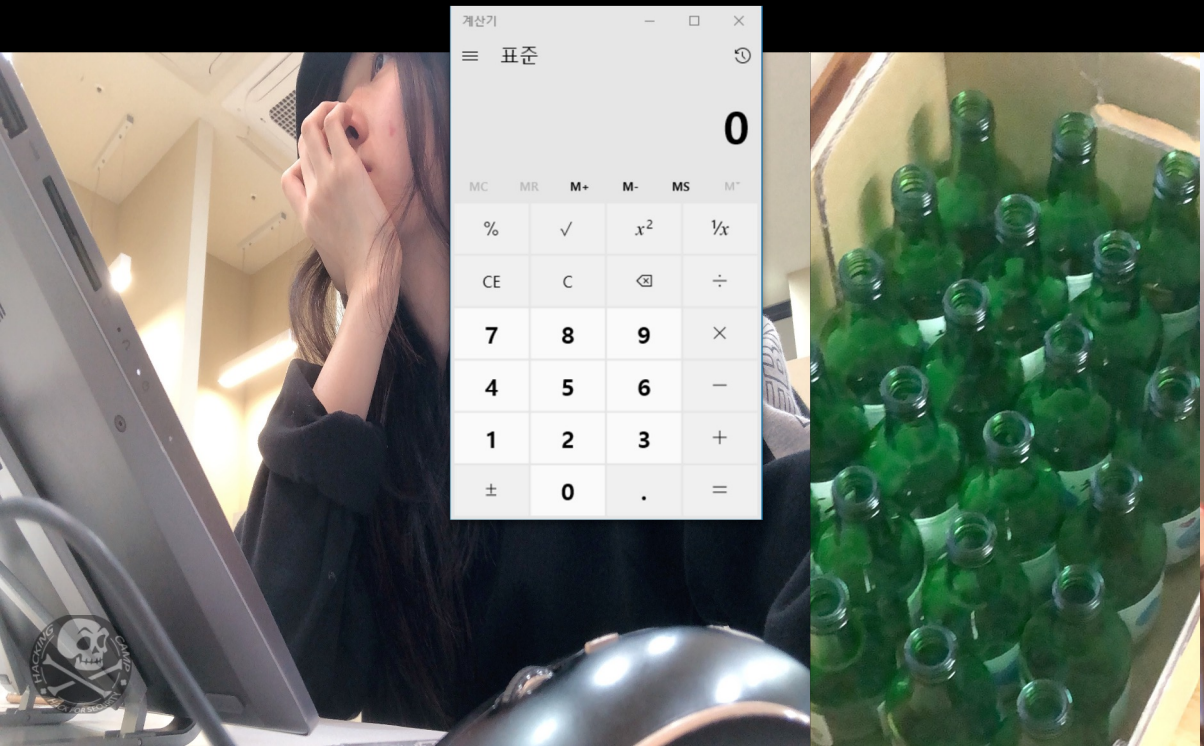
“내 코드는 안전하고 버그가 없을거야"



"내 코드는 안전하고 버그가 없을거야"



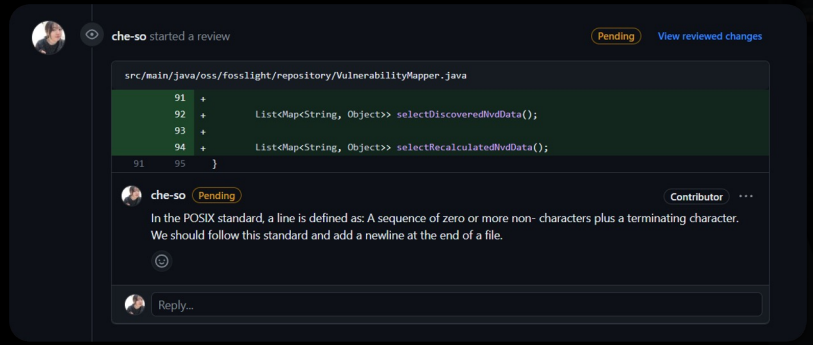
~~"내 코드는 안전하고 버그가 없을거야"~~



코드리뷰 & 테스트

충분한 코드 검토와 테스트를 거쳐야 안전하다.

Pull Request(PR)시 코드리뷰 수행



코드리뷰 & 테스트

충분한 코드 검토와 테스트를 거쳐야 안전하다.

단위 테스트(Unit Test) 작성

```python
from ruleset.definition import *

import unittest

class TEST(unittest.TestCase):
    def test_ipv4(self):
        tmp = {
            "ipv4Network": "10.0.0.0/8"
        }
        r = IPv4Network("10.0.0.0/8")
        self.assertEqual(Definition._ipv4(tmp, "ipv4Network"), r)

    def test_ipv6(self):
        tmp = {
            "ipv6Network": "::1"
        }
        r = IPv6Network("0::1")
        self.assertEqual(Definition._ipv6(tmp, "ipv6Network"), r)

    def test_str(self):
        tmp = {
            "method": "PUT"
        }
        self.assertEqual(Definition._str(tmp, "method"), "PUT")

    def test_group(self):
        group = {
            "a": "b"
        }
        tmp = {
            "group": group
        }
        self.assertEqual(Definition._group(tmp, "group"), group)

    def test_header(self):
        header = {
            "User-Agent": ".*"
        }
        tmp = {
            "header": header
        }
        result = {
            "user-agent": ".*"
        }
        self.assertEqual(Definition._header(tmp, "header"), result)
```

코드리뷰 & 테스트

충분한 코드 검토와 테스트를 거쳐야 안전하다.

CI/CD 파이프라인에서 정적 분석과 퍼징을 수행

AFL++ OWASP ZAP Semgrep

안전한 언어와 프레임워크 사용

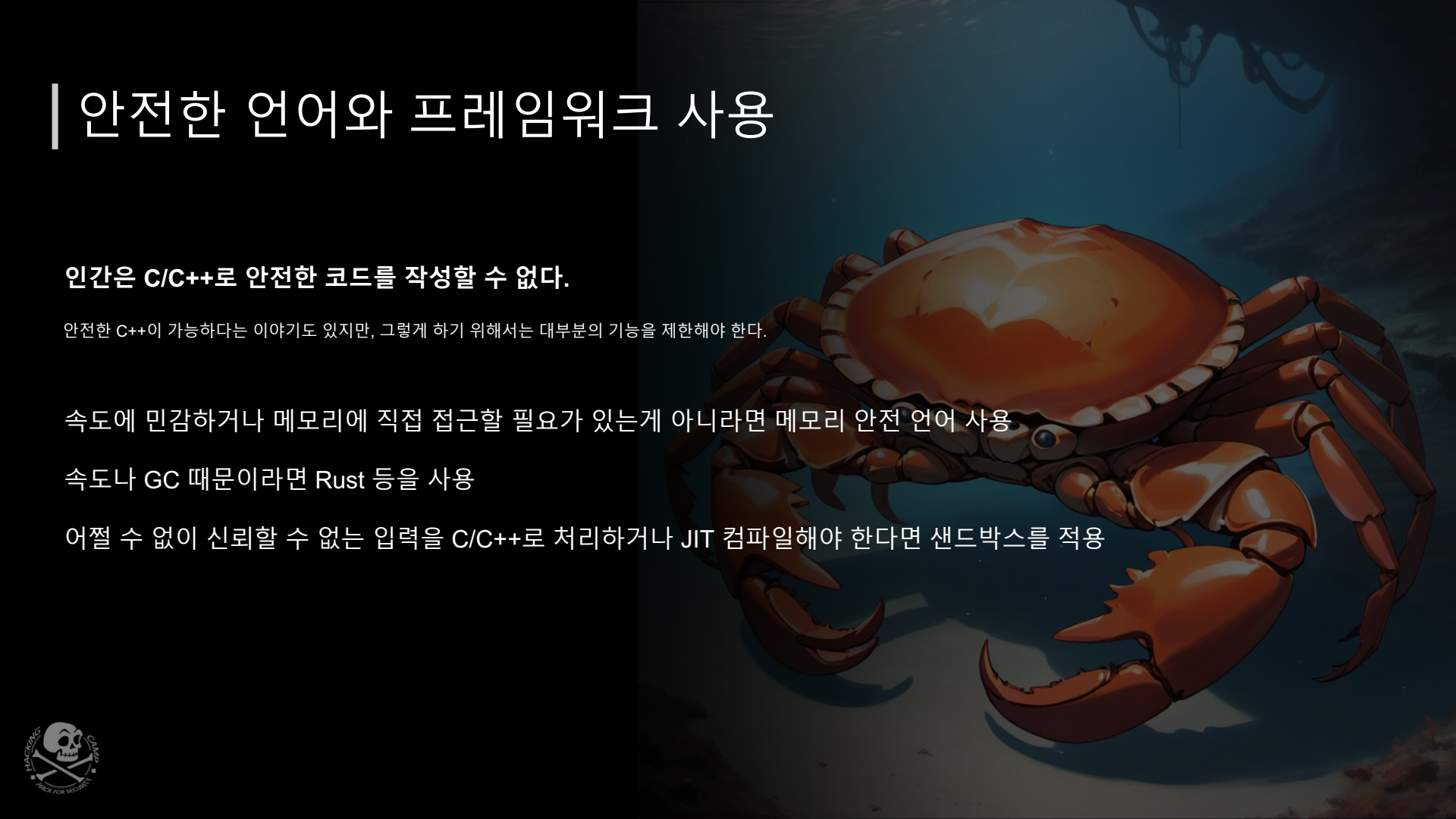
인간은 C/C++로 안전한 코드를 작성할 수 없다.

안전한 C++이 가능하다는 이야기도 있지만, 그렇게 하기 위해서는 대부분의 기능을 제한해야 한다.

속도에 민감하거나 메모리에 직접 접근할 필요가 있는게 아니라면 메모리 안전 언어 사용

속도나 GC 때문이라면 Rust 등을 사용

어쩔 수 없이 신뢰할 수 없는 입력을 C/C++로 처리하거나 JIT 컴파일해야 한다면 샌드박스를 적용

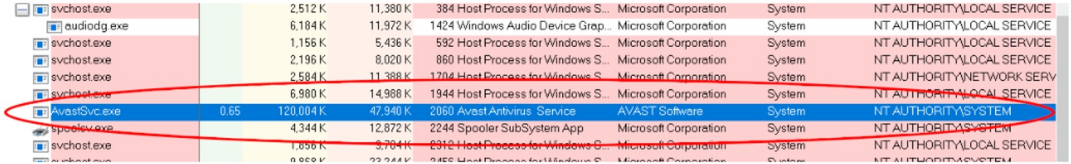


불필요하게 복잡한 기능 제거

Avast Antivirus JavaScript Interpreter

> NOTE: On 03/11/2020 Avast announced they had decided to disable this interpreter globally!

The main Avast antivirus process is called AvastSvc.exe, which runs as SYSTEM.



svchost.exe		2,512 K	11,380 K	384 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
audiodg.exe		6,184 K	11,972 K	1424 Windows Audio Device Grap...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
svchost.exe		1,156 K	5,436 K	592 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
svchost.exe		2,196 K	8,020 K	860 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
svchost.exe		2,584 K	11,388 K	1704 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\NETWORK SERV
svchost.exe		6,980 K	14,988 K	1944 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
AvastSvc.exe	0.65	120,004 K	47,940 K	2060 Avast Antivirus Service	AVAST Software	System	NT AUTHORITY\SYSTEM
spoolsv.exe		4,344 K	12,872 K	2244 Spooler SubSystem App	Microsoft Corporation	System	NT AUTHORITY\SYSTEM
svchost.exe		1,656 K	5,764 K	2312 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE

That service loads the low level antivirus engine, and analyzes untrusted data received from sources like the filesystem minifilter or intercepted network traffic.

Despite being highly privileged and processing untrusted input by design, it is unsandboxed and has poor mitigation coverage. Any vulnerabilities in this process are critical, and easily accessible to remote attackers.

So.. maybe not great that it includes a custom JavaScript interpreter....???? 🙃

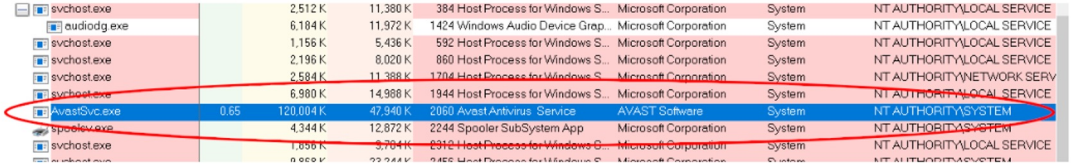
github.com/taviso/avscript

불필요하게 복잡한 기능 제거

Avast Antivirus JavaScript Interpreter

> NOTE: On 03/11/2020 Avast announced they had decided to disable this interpreter globally!

The main Avast antivirus process is called AvastSvc.exe, which runs as SYSTEM.



svchost.exe		2,512 K	11,380 K	384 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
audiodg.exe		6,184 K	11,972 K	1424 Windows Audio Device Grap...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
svchost.exe		1,156 K	5,436 K	592 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
svchost.exe		2,196 K	8,020 K	860 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
svchost.exe		2,584 K	11,388 K	1704 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\NETWORK SERV
svchost.exe		6,980 K	14,988 K	1944 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE
AvastSvc.exe	0.65	120,004 K	47,940 K	2060 Avast Antivirus Service	AVAST Software	System	NT AUTHORITY\SYSTEM
spoolsv.exe		4,344 K	12,872 K	2244 Spooler SubSystem App	Microsoft Corporation	System	NT AUTHORITY\SYSTEM
svchost.exe		1,656 K	5,764 K	2312 Host Process for Windows S...	Microsoft Corporation	System	NT AUTHORITY\LOCAL SERVICE

That service loads the low level antivirus engine, and analyzes untrusted data received from sources like the filesystem minifilter or intercepted network traffic.

Despite being highly privileged and processing untrusted input by design, it is unsandboxed and has poor mitigation coverage. Any vulnerabilities in this process are critical, and easily accessible to remote attackers.

So.. maybe not great that it includes a custom JavaScript interpreter....???? 🙃

github.com/taviso/avscript

버그클래스 단위로 패치

chromium:40063132 (CVE-2023-1215)

```
--- a/third_party/blink/renderer/core/css/cssom/style_property_map.cc
+++ b/third_party/blink/renderer/core/css/cssom/style_property_map.cc

@@ -381,6 +381,17 @@
         "Cannot append to a list containing a variable reference");
       return;
     }
+    if (!css_value->IsValueList()) {
+      // The standard doesn't seem to cover this explicitly
+      // (https://github.com/w3c/css-houdini-drafts/issues/823),
+      // but the only really reasonable solution seems to be
+      // to throw a TypeError.
+      //
+      // This covers e.g. system-wide CSS keywords, like inherit.
+      exception_state.ThrowTypeError(
+          "Cannot append to something that is not a list");
+      return;
+    }
     current_value = To<CSSValueList>(css_value)->Copy();
   } else {
     current_value = CssValueListForPropertyID(property_id);
```

chromium:40062700

```
--- a/third_party/blink/renderer/core/css/cssom/style_property_map.cc
+++ b/third_party/blink/renderer/core/css/cssom/style_property_map.cc

@@ -373,6 +373,13 @@

   CSSValueList* current_value = nullptr;
   if (const CSSValue* css_value = GetProperty(property_id)) {
+    if (css_value->IsVariableReferenceValue()) {
+      // https://drafts.css-houdini.org/css-typed-om/#dom-stylepropertymap-append
+      // 8. If props[property] contains a var() reference, throw a TypeError.
+      exception_state.ThrowTypeError(
+          "Cannot append to a list containing a variable reference");
+      return;
+    }
     current_value = To<CSSValueList>(css_value)->Copy();
   } else {
     current_value = CssValueListForPropertyID(property_id);
```

버그클래스 단위로 패치

chromium:40063132 (CVE-2023-1215)

```
--- a/third_party/blink/renderer/core/css/cssom/style_property_map.cc
+++ b/third_party/blink/renderer/core/css/cssom/style_property_map.cc

@@ -381,6 +381,17 @@
           "Cannot append to a list containing a variable reference");
       return;
     }
+    if (!css_value->IsValueList()) {
+      // The standard doesn't seem to cover this explicitly
+      // (https://github.com/w3c/css-houdini-drafts/issues/823),
+      // but the only really reasonable solution seems to be
+      // to throw a TypeError.
+      //
+      // This covers e.g. system-wide CSS keywords, like inherit.
+      exception_state.ThrowTypeError(
+          "Cannot append to something that is not a list");
+      return;
+    }
     current_value = To<CSSValueList>(css_value)->Copy();
   } else {
     current_value = CssValueListForPropertyID(property_id);
```

chromium:40062700

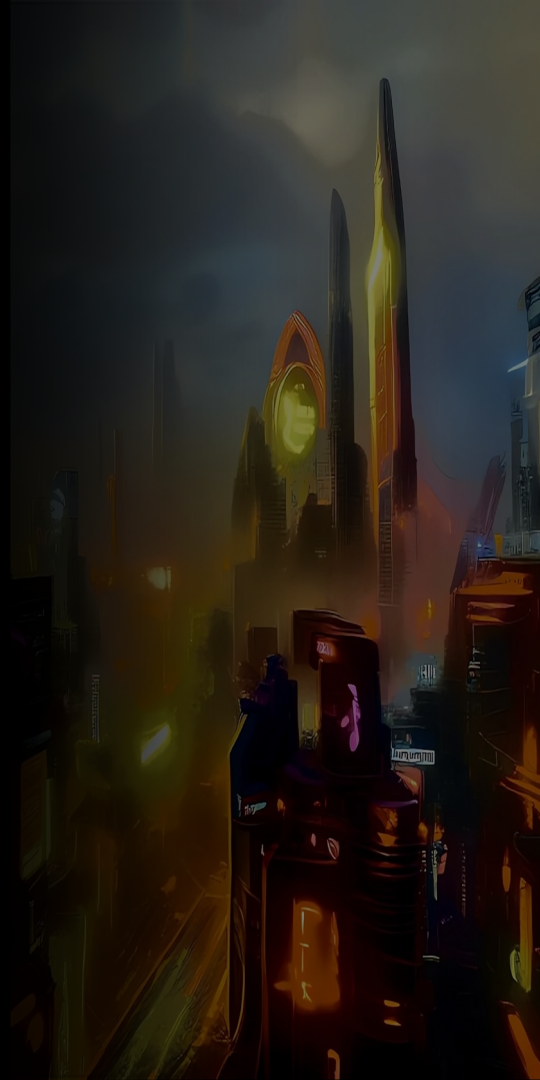
```
--- a/third_party/blink/renderer/core/css/cssom/style_property_map.cc
+++ b/third_party/blink/renderer/core/css/cssom/style_property_map.cc

@@ -373,6 +373,13 @@

   CSSValueList* current_value = nullptr;
   if (const CSSValue* css_value = GetProperty(property_id)) {
+    if (css_value->IsVariableReferenceValue()) {
+      // https://drafts.css-houdini.org/css-typed-om/#dom-stylepropertymap-append
+      // 8. If props[property] contains a var() reference, throw a TypeError.
+      exception_state.ThrowTypeError(
+          "Cannot append to a list containing a variable reference");
+      return;
+    }
     current_value = To<CSSValueList>(css_value)->Copy();
   } else {
     current_value = CssValueListForPropertyID(property_id);
```

Type Confusion

버그클래스 단위로 패치

```
--- a/third_party/blink/renderer/platform/wtf/casting.h
+++ b/third_party/blink/renderer/platform/wtf/casting.h

@@ -114,13 +114,12 @@
   return from && IsA<Derived>(*from);
 }

-// Unconditionally downcasts from Base to Derived. Internally, this asserts
-// that |from| is a Derived to help catch bad casts in testing/fuzzing. For the
-// pointer overloads, returns nullptr if the input pointer is nullptr.
+// Unconditionally downcasts from Base to Derived. Internally, this asserts that
+// |from| is a Derived to help catch bad casts. For the pointer overloads,
+// returns nullptr if the input pointer is nullptr.
 template <typename Derived, typename Base>
 const Derived& To(const Base& from) {
-  // TODO(https://crbug.com/1448838): Upgrade this to a CHECK.
-  SECURITY_DCHECK(IsA<Derived>(from));
+  CHECK(IsA<Derived>(from));
   return static_cast<const Derived&>(from);
 }

@@ -131,8 +130,7 @@

 template <typename Derived, typename Base>
 Derived& To(Base& from) {
-  // TODO(https://crbug.com/1448838): Upgrade this to a CHECK.
-  SECURITY_DCHECK(IsA<Derived>(from));
+  CHECK(IsA<Derived>(from));
   return static_cast<Derived&>(from);
 }
 template <typename Derived, typename Base>
```

SECURITY_DCHECK → CHECK



미티게이션 적용

MTE **PAC**

DEP/NX **Sandbox**

ASLR **Stack Canary**

개발자와 보안 담당자가 이런 노력을 충분히 하고 있으니까

버그나 취약점이 없고 안전하겠지?

자원은 한정적이다

속도, 안정성, UX, 유지보수, 보안 모두 중요하지만..

개발에 쓸 수 있는 시간, 예산, 인력 등은 유한하다.

코드 검토와 퍼징에 쓸 수 있는 자원은 제한적이고, 언어와 프레임워크는 개발자의 편의성도 보장해야 한다.

실행할 시스템의 CPU, 메모리 등은 유한하다.

이미 검증한 값을 계속해서 다시 검증할 수 없고, 미티게이션을 무한정 추가할 수 없다.

자원은 한정적이다

속도, 안정성, UX, 유지보수, 보안 모두 중요하지만..

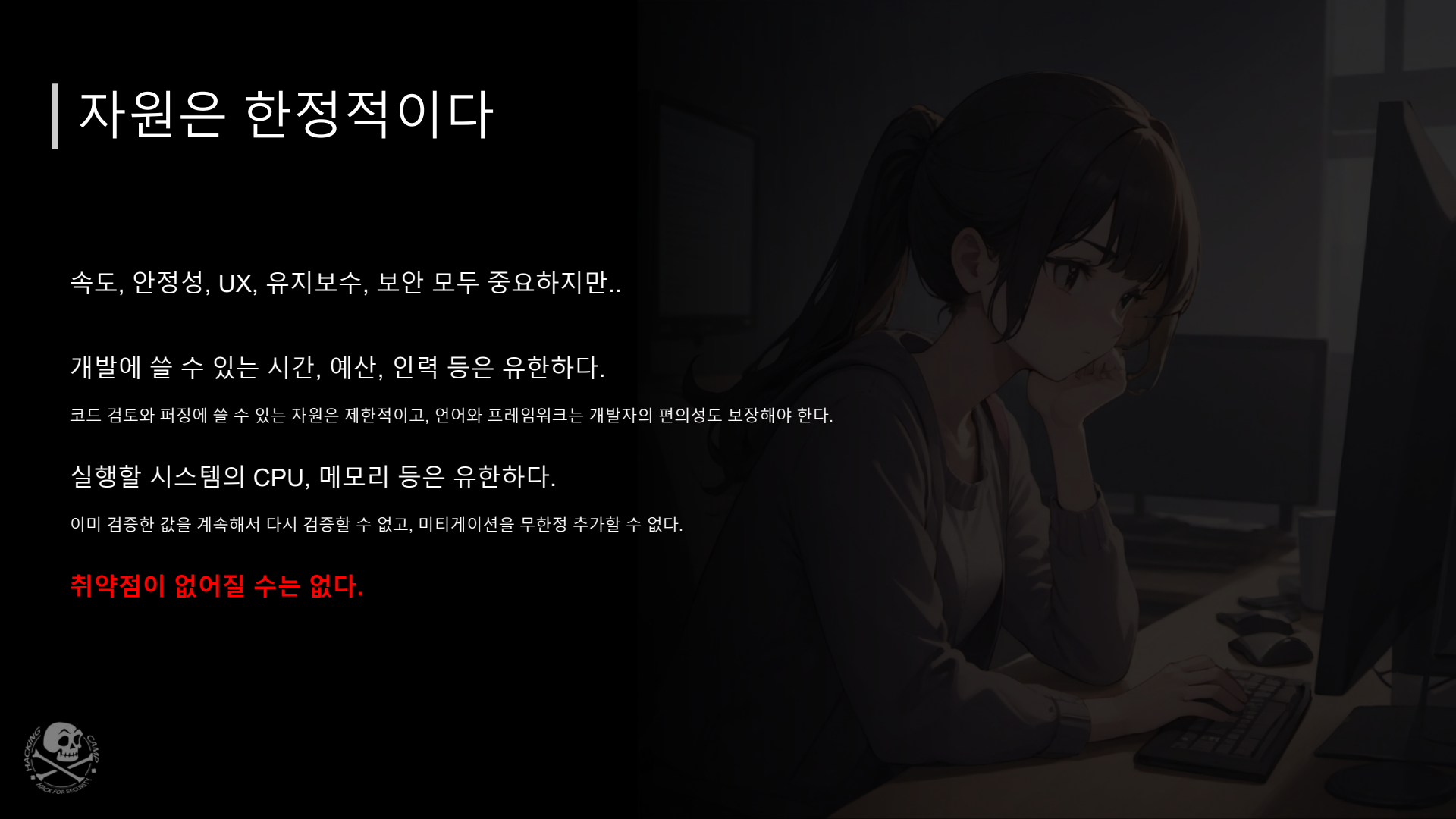
개발에 쓸 수 있는 시간, 예산, 인력 등은 유한하다.

코드 검토와 퍼징에 쓸 수 있는 자원은 제한적이고, 언어와 프레임워크는 개발자의 편의성도 보장해야 한다.

실행할 시스템의 CPU, 메모리 등은 유한하다.

이미 검증한 값을 계속해서 다시 검증할 수 없고, 미티게이션을 무한정 추가할 수 없다.

취약점이 없어질 수는 없다.



개발자와 해커의 편견

“이 기능에서는 이미 취약점이 여러번 나왔으니까 안전할거야”

“누가 우리 회사를 공격하겠어”

“이건 취약점이 아니라 정상 기능이야”

“Rust로 작성되었으니까 메모리 버그는 없을거야”

“React로 만들었으니까 XSS 취약점은 없겠지”

“이 코드는 충분히 퍼징되었으니 취약점이 없을거야”

Security Mistakes

개발자들이 자주 하는 보안 실수

Security Mistakes

개발자들이 어쩔 수 없이 자주 하는 보안 실수

개발자에게 물어보기로 했습니다

"개발자들이 자주 하는 보안 실수" 이런 주제 생각하고 있는데

누나 혹시 해킹캠프에서 같이 발표하실 생각 있으신가요?

협업 프로젝트를 진행하면서 평소 제가 (제일) 많이 실수하기도 하고, 주변 개발자가 실수하는걸 많이 보기도 했는데... 해킹캠프에서 발표해서 해커분들과 개발자분들께 제 경험을 나눌 수 있다면 저는 너무 좋은 것 같아요!

Case 1 - IDOR 실수

Django로 대외활동 팀원 모집 프로젝트 제작 중 해당 문제 발생

Team 모델에서는 다른 모델의 인스턴스에 대한 참조를 포함하고 있다.

하지만 TeamDetailAPIView에서는 팀의 세부 정보를 가져오거나 수정할 때 권한 검사가 이루어지지 않고 있다. 따라서 아래와 같은 요청을 보낼 시 원래는 불가능해야 하는 일인, 권한이 없는 사용자가 다른 사용자의 팀, 팀원, 또는 신청 정보에 접근/수정하는 행위가 가능해진다.

```
GET /api/contest/{contestPk}/team/{teamPk}/
```

```python
class Team(models.Model):
    name = models.CharField(max_length=50)
    teamname = models.CharField(max_length=100,null=True)
    call = models.CharField(max_length=100,null=True)
    detail = models.TextField(null=True)
    tendency = models.TextField(default="[]", null= True)
    contest = models.ForeignKey(Contest, on_delete=models.CASCADE)
    created_by = models.ForeignKey(User, on_delete=models.CASCADE, null=True)
    dev_capacity = models.PositiveIntegerField(default=0)
    plan_capacity = models.PositiveIntegerField(default=0)
    design_capacity = models.PositiveIntegerField(default=0)
    dev = models.PositiveIntegerField(default=0)
    plan = models.PositiveIntegerField(default=0)
    design = models.PositiveIntegerField(default=0)
    jickgoon_type = models.CharField(max_length=50, blank=True, choices=[('dev', '개발'), ('plan', '기획'), ('design', '디자인')])

    def get_tendency_as_list(self):
        return json.loads(self.tendency)

class Member(models.Model):
    team = models.ForeignKey(Team, on_delete=models.CASCADE, related_name = "members")
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    jickgoon = models.CharField(max_length=50, choices=[('dev', '개발'), ('plan', '기획'), ('design', '디자인')])

class Application(models.Model):
    team = models.ForeignKey(Team, on_delete=models.CASCADE)
    applicant = models.ForeignKey(User, on_delete=models.CASCADE)
    jickgoon = models.CharField(max_length=50, choices=[('dev', '개발'), ('plan', '기획'), ('design', '디자인')])
    is_approved = models.BooleanField(default=False)
```

```python
#팀 세부페이지
class TeamDetailAPIView(APIView):
    #팀 세부페이지 가져오기
    def get(self, request, teamPk, contestPk):
        team = get_object_or_404(Team, pk=teamPk)
        members = Member.objects.filter(team=team)
        dev_members = members.filter(jickgoon='dev')
        plan_members = members.filter(jickgoon='plan')
        design_members = members.filter(jickgoon='design')

        dev_member_data = [{'user': member.user.username} for member in dev_members]
        plan_member_data = [{'user': member.user.username} for member in plan_members]
        design_member_data = [{'user': member.user.username} for member in design_members]

        team_serializer = TeamSerializer(team)
        data = {
            **team_serializer.data,
            'dev_members': dev_member_data,
            'plan_members': plan_member_data,
            'design_members': design_member_data,
        }

        return Response(data)
```

Case 2 - Race Condition으로 인한 개발 실수

Django로 대외활동 팀원 모집 프로젝트 제작 중 해당 문제 발생

사용자 A가 공모전 정보 창에서 스크랩 버튼을 클릭 시, 스크랩 한 사람 수를 표시하기 위해 내부 모델에서 값에 +1을 추가하여 업데이트하도록 설정하였으나, 다수의 유저가 사용시 한 스레드가 처리되는 동안 다른 스레드가 같은 함수에 들어와 작업이 처리되는 상황이다.

➠

레이스 컨디션으로 스크랩한 사용자 수 표시 오류 발생,

여러명이 접근시 검증/수정 과정에서의 데이터 차이로 인해 피해 발생 가능성

```
post = Post.objects.get(id=1)
post.scrap_count = post.scrap_count + 1
post.save()
```

Case 3 - GitHub 관리 실수

Django로 대외활동 팀원 모집 프로젝트 제작 중 해당 문제 발생

비밀 키(SECRET_KEY)는 Django에서 암호화, 세션 및 기타 보안 메커니즘에 사용되므로 절대로 공개하지 말아야 하지만, 초기 설정 중 .gitignore를 사용하지 않고 GitHub에 올려서 커밋 히스토리에 비밀 키가 공개되어 있다.

잘못 커밋한 경우 커밋 히스토리에서도 삭제해야 한다.

practice2 / myproject / settings.py

Code | Blame | 129 lines (98 loc) · 3.47 KB | Your organization can pay for GitHub Copilot

```
8
9   For the full list of settings and their values, see
10  https://docs.djangoproject.com/en/4.1/ref/settings/
11  """
12
13  from pathlib import Path
14  import os
15  # Build paths inside the project like this: BASE_DIR / 'subdir'.
16  BASE_DIR = Path(__file__).resolve().parent.parent
17
18
19  # Quick-start development settings - unsuitable for production
20  # See https://docs.djangoproject.com/en/4.1/howto/deployment/checklist/
21
22  # SECURITY WARNING: keep the secret key used in production secret!
23  SECRET_KEY = 'django-insecure
24
25  # SECURITY WARNING: don't run with debug turned on in production!
26  DEBUG = True
27
28  ALLOWED_HOSTS = ["*"]
29
30
31  # Application definition
32
33  INSTALLED_APPS = [
34      'django.contrib.admin',
35      'django.contrib.auth',
36      'django.contrib.contenttypes',
37      'django.contrib.sessions',
38      'django.contrib.messages',
39      'django.contrib.staticfiles',
```

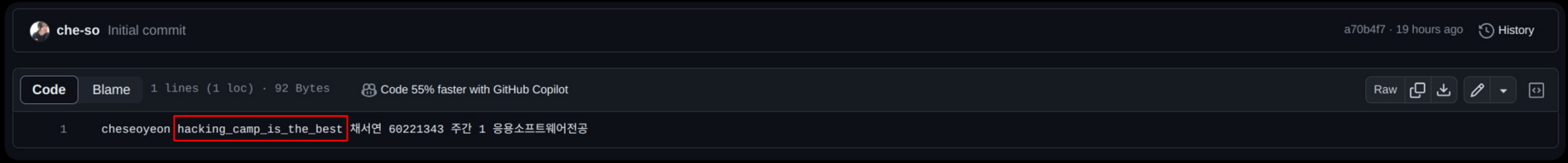
Quiz 채서연의 비밀번호는?

Quiz 채서연의 비밀번호는?

Hint github.com/che-so

Quiz 채서연의 비밀번호는?

원래 진짜 비밀번호가 올라가 있었는데, 발표 준비하다가 알게 되어서 변경 후 퀴즈에 출제했다.



github.com/che-so/2022-2_Final/blob/master/account/account

Q&A

Thank You

HACKINGCAMP 28

개발자들이 자주 하는 보안 실수


instagram.com/che._.so


instagram.com/movptr

This presentation uses pixabay, carbon, wrtn, stable diffusion and flaticon.