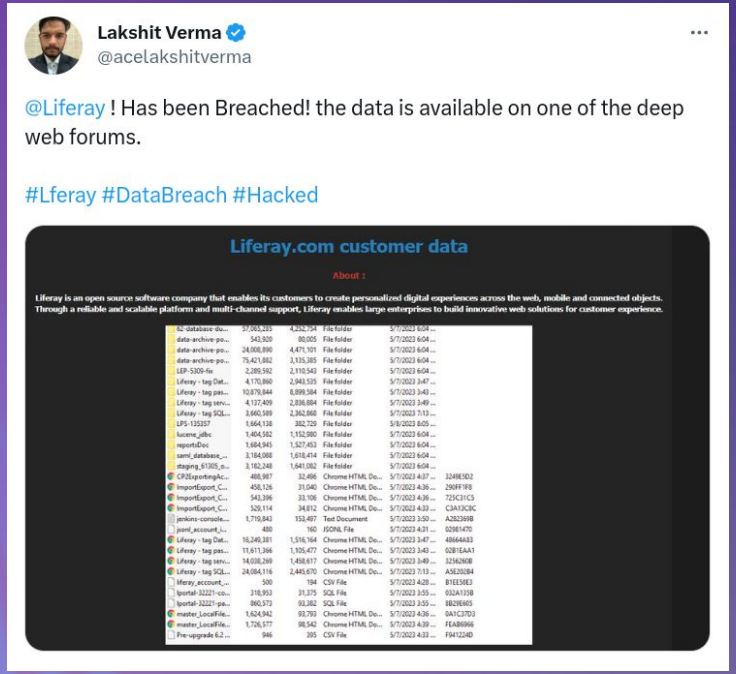
Liferay® DEVCON

Martin Žember and Attila Danko

Securing your Liferay: a 360 approach

Protect your Liferay!

Liferay Breached?



Liferay® DEVCON

How To Hack Easily

And How To Defend

The ways we usually hack (ethically)

- **Password guessing**
  - Default passwords
  - Spraying
  - Dictionary attacks

Spraying

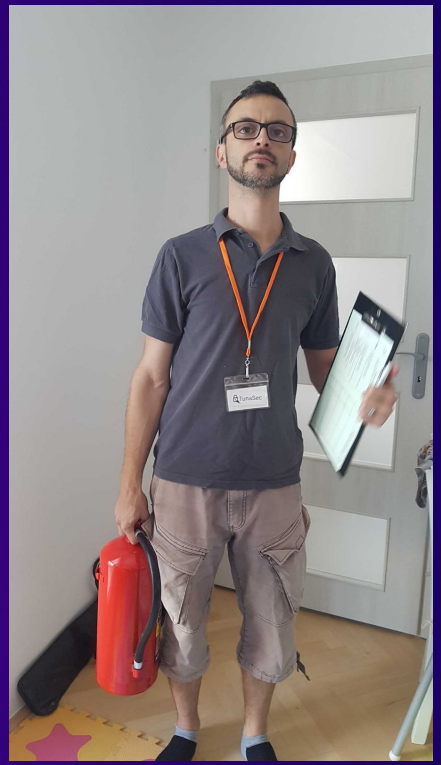
The ways we usually hack (ethically)

- Password guessing
  - Default passwords
  - Spraying
  - Dictionary attacks
- **Vulnerabilities**

The ways we usually hack (ethically)

- Password guessing
  - Default passwords
  - Spraying
  - Dictionary attacks
- Vulnerabilities
- **Physical breaches**

Physical intrusion: night & day

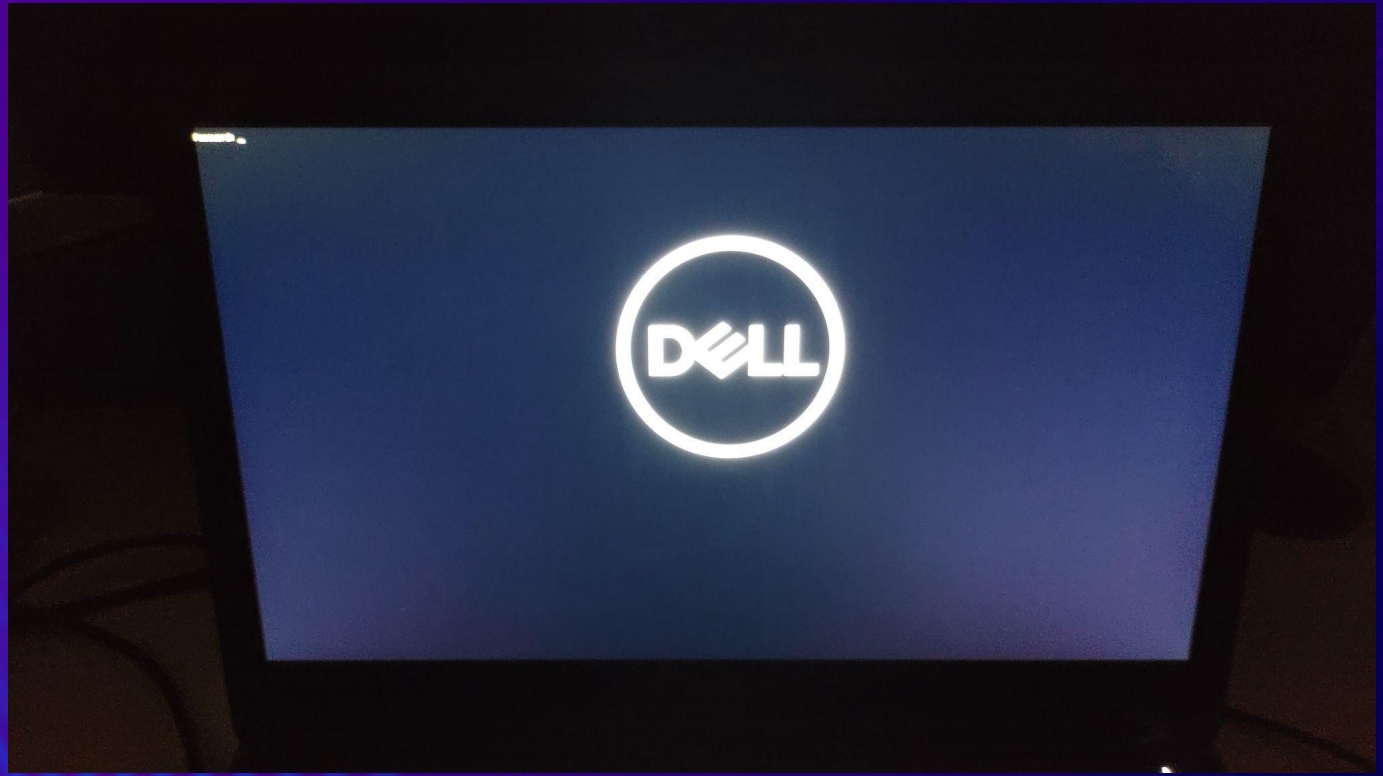


Physical intrusion: bypassing locks

Physical intrusion: safes that are not safes

Physical intrusion: safes that are not safes

Physical intrusion: encrypted laptop



The ways we usually hack (ethically)

- Password guessing
  - Default passwords
  - Spraying
  - Dictionary attacks
- Vulnerabilities
- Physical breaches
- **Phishing**

The ways we usually hack (ethically)

- Password guessing
  - Default passwords
  - Spraying
  - Dictionary attacks
- Vulnerabilities
- Physical breaches
- Phishing
- **Discovering credentials**

The ways we usually hack (ethically)

- Password guessing
  - Default passwords
  - Spraying
  - Dictionary attacks
- Vulnerabilities
- Physical breaches
- Phishing
- **Discovering credentials**
  - Your git (source that contains secrets)

The ways we usually hack (ethically)

- Password guessing
  - Default passwords
  - Spraying
  - Dictionary attacks
- Vulnerabilities
- Physical breaches
- Phishing
- **Discovering credentials**
  - Your git (source that contains secrets)
  - Leaks (hacked database or endpoint dumps sold or shared)

The ways we usually hack (ethically)

- Password guessing
  - Default passwords
  - Spraying
  - Dictionary attacks
- Vulnerabilities
- Physical breaches
- Phishing
- **Discovering credentials**
  - Your git (source that contains secrets)
  - Leaks (hacked database or endpoint dumps sold or shared)
  - Once inside: shared credentials among users

Defense against attacks

- Password guessing
  - Strong passwords
- Vulnerabilities
  - Filtering ports by default
  - Updating the software
- Physical breaches
  - Encryption of drives
  - Alarms
- Phishing
  - Password manager
    - One that does not fill your password to e.g. https://www.iinkedin.com
  - Education
- Discovering credentials
  - Your git secrets: deploy gitleaks (or truffleHog)
  - Leaks of hacked DBs: haveibeenpwned.com
  - Once inside: Search for ‘password’ shared in your company Slack, Google Drive, …

Workshop demo #1: Liferay's open port 8080

- Port 8080 is open to everybody on the local network (by default)
- Attacker finds it
  - sudo nmap -T4 192.168.25.195/24 -sS -p8080 -sV -vvv
- Attacker opens Groovy console and runs a reverse shell

Workshop task #1: filter your ports

UFW:

1. sudo ufw enable
2. sudo ufw allow 22

To show the settings:

3. sudo ufw status verbose
4. sudo ufw status numbered

Alternatively, **iptables:**

5. sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
6. sudo iptables -P INPUT DROP

To show the settings:

7. sudo iptables -L

iptables forgets **after reboot.**
Solution:

1. sudo apt-get install iptables-persistent
2. sudo service iptables-persistent start

If you run ufw and want to get back to iptables, iptables -L becomes very long. To **reset:**

3. sudo iptables-restore < /etc/iptables/rules.v4

Liferay® DEVCON

Mod_security / WAF

Protect your Liferay!

Protect your Liferay!

Where your Liferay instance operate?

On-premise / local -> use MOD security - applied on Nginx

Cloud -> WAF (Web Application Firewall) - applied on ingress traffic or Load Balancers

FYI: solutions are using OWASP TOP 10 framework!

Why?

- Prevent typical attacks (SQL injections, cross-site scripting (XSS), and distributed denial of service (DDoS))
- Save resources (prevented malicious request doesn't use your resources) Your cloud provider charge you based on resource usage!
- Ensure compliance (security tools are required)

Protect your Liferay!

Security responsibility matrix

	Liferay on-premise / self-hosted	Liferay LXC-SM (former DXP Cloud)	Liferay LXC (Liferay Experience Cloud)
Customer	Full responsibility	You're responsible for your own data! Shared - PaaS model	Client site data, User and access management
Liferay	No responsibility	monitoring / alert customers in major cases	N/A
Liferay SOC (Security Operation Center)	No responsibility	Not available	Monitor / alert / prevent / 0-24 service

Protect your Liferay!

Why me? I'm a developer!

- Security is everybody's responsibility (especially on their own field)!
- Your code/app need to cooperate with security tools!

  -> you need to be familiar with OWASP TOP 10 otherwise valid request be drop!

Protect your Liferay!

WAF / MOD sec in action:

- Core security rules sets (static): https://github.com/SpiderLabs/ModSecurity

Sample rule: /rules/REQUEST-913-SCANNER-DETECTION.conf

```
# -=[ Crawler User-Agents ]=-
#
# This rule detects user-agents associated with various crawlers, SEO tools,
# and bots, which have been reported to potentially misbehave.
# These crawlers can have legitimate uses when used with authorization.
#
# This rule is a sibling of rule 913100.
#
SecRule REQUEST_HEADERS:User-Agent "@pmFromFile crawlers-user-agents.data" \
    "id:913102,\
```

Protect your Liferay!

WAF / MOD sec in action:

- Cloud Armor preconfigured rules: https://cloud.google.com/armor/docs/waf-rules

SQL injection
Cross-site scripting
Local file inclusion
Remote file inclusion
Remote code execution
Method enforcement
Scanner detection
Protocol attack
PHP injection attack
Session fixation attack
Java attack
NodeJS attack

Protect your Liferay!

WAF / MOD sec in action:

- Security rules sets

Paranoia level:

PL 1: Baseline Security with a minimal need to tune away false positives. This is CRS for everybody running an HTTP server on the internet.

PL 2: Rules that are adequate when real customer data is involved. Expect false positives and learn how to tune them away.

PL 3: Online banking level security with lots of false positives. From a project perspective, false positives are accepted here, so you need to be able to help yourself by writing rule exclusions.

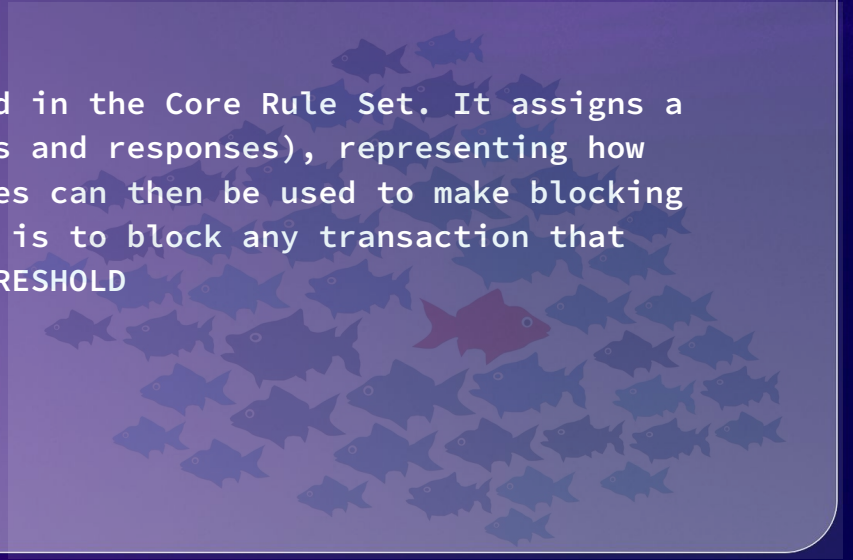
PL 4: Rules that are so strong (or paranoid) they are adequate to protect the crown jewels. Use at your own risk and be prepared to get a large number of false positives.

Protect your Liferay!

WAF / MOD sec in action:

- Anomaly detection (dynamic)

Anomaly scoring, is a scoring mechanism used in the Core Rule Set. It assigns a numeric score to HTTP transactions (requests and responses), representing how ‘anomalous’ they appear to be. Anomaly scores can then be used to make blocking decisions. The default CRS blocking policy, is to block any transaction that meets or exceeds a defined anomaly score THRESHOLD



Protect your Liferay!

WAF / MOD sec in action:

DEMO time!!!

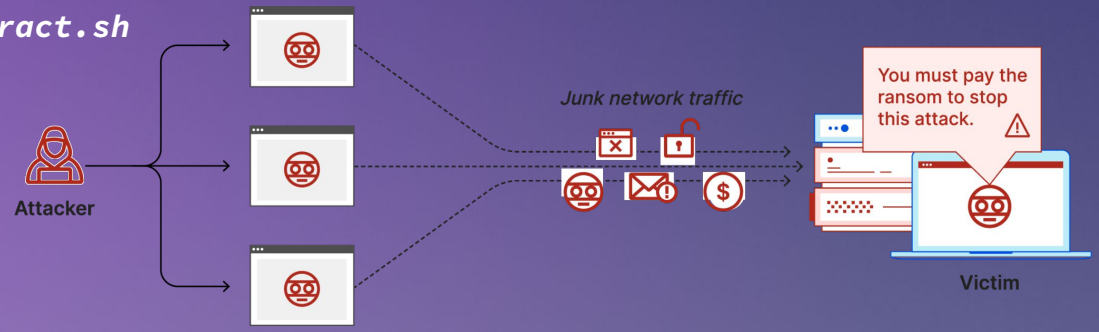
Protect your Liferay!

Some thoughts about DDOS attacks

- It’s https traffic now!
- Typical attack pattern changed (burst - short time with big amount of data)

  Sorry but scaling doesn't help! :(

- Use for hide other attack vectors
- Attackers use valid testing tools like ***interact.sh***
- Ransom DDOS attack



Protect your Liferay!

WAF / MOD sec in action:

- DDOS protection

Multiple solution on the market provides DDOS protection service

LXC use Cloud Armor against DDOS attacks:

Google Cloud Armor gets deployed along with the load balancer and provides protection against Network – L3/L4 Attacks, Application – L7 and Protocol based Volumetric Attacks. It mitigates network attacks and **only allows well formed packets through the load balancing proxies.**

Real Time alerts are sent to Cloud Monitoring for viewing and information about traffic spikes etc are sent to **Security Command Centre** for investigation

Liferay® DEVCON

How They Hack

And How To Defend Easily

Ways to hack, specifically by blackhats (criminals)

- Malware infection
  - Trojan horses (in warez like MS Office)
  - Cracks, keygens
  - Malware deploys Information Stealers
  - …
    - Defense:
      - Awareness
      - Antivirus / EDR
      - Blue team
- Malicious dependencies
  - Dependency confusion attacks
  - Typosquatting
  - Hacking existing repository owners

Malicious dependencies: defense

- Peer review
  - Dependency infects the developer but not the production
- Known dependencies
  - Spreadsheet with dependencies
  - Nexus repository
- Environment for development
  - Liferay Workspace, deps are included
- Virtualization, e.g. Linux Containers (LXC)
  - Browser running in LXC: isolates from the development and vice versa
- Password manager, SSH keys in 1Password
- Linux
  - Amount of malware is smaller
- Antivirus / EDR

Liferay Security 360 checklist for home exercise

- Developers
  - Hiring background checks
  - Security trainings
  - Secured laptops, repositories, credentials+MFA
- Product Development & Release
  - Secure SDLC based on OWASP + SW licence and data privacy compliance
  - Code/product vulnerability scans (SAST, DAST, deps)
  - Standardized dev & build environment and tooling
  - Secured CI and Release environments
- Live Systems
  - Automated deployments and manual verification
  - Security & Site Reliability Monitoring
  - DDoS, WAF, IDS, Antivirus, internal protections
  - Internal and external security testing
- Vulnerability & Patch Management
  - Internal SLAs with teams, CVE with CVSS for all vulnerabilities
  - Internal AppSec and InfoSec teams to help
  - Vulnerability Disclosure Program for External researchers

Summary

How To Hack Easily

- Passwords
- Vulnerable services
- …

Firewall

- Filter open ports

WAF (mod_security)

- Encourage you to use WAF or mod_security
- Be aware DDOS
- Cooperate with security, we have the same goal

How They Hack

- Malware
- Malicious dependencies
- Checklist

How was this session?

Please share your rating in
the event app. Thank you!