



An Attacker Looks At Docker

Approaching Multi-Container Applications

Wesley McGrew, Ph.D.

Director of Cyber Operations

HORNE Cyber

wesley.mcgrew@hornecyber.com @mcgrewsecurity



My Background

- Ph.D. Computer Science – Mississippi State University
- Academia
 - NSA CAE – Research, Education, Cyber Operations – MSU
 - Industrial Control Systems – Human-Machine Interfaces
 - Research & Education - Reverse Engineering & Malware Attribution
- Private
 - Director of Cyber Operations – HORNE Cyber
 - Computer Network Operations – CNO/CNE/CAN
 - Penetration testing, red teaming, application security
 - Operational security of testing engagements



Intentions

- ...to make a strong point about the relationship between an attacker's skill set (and its development over time) vs. developer trends.
 - How to leverage what you already know
 - How to look at learning new technologies moving forward
- ...to provide a hacker experienced in exploitation and post-exploitation of **networks of systems** an exposure to applications composed of **multiple containers**
 - Exploring application internals
- ...with concrete Docker examples that leverage common practices (those in tutorials and intuitive/naïve usage)
- Inspiration concept/approach –
HD Moore/Valsmith DEF CON 15 *Tactical Exploitation*
- Target audience – Attackers – Pentest, Red Team, CNE, CNO



Prior Art in Docker

- David Mortman, *Docker, Docker, Give Me the News, I Got a Bad Case of Securing you*, DEF CON 23
 - Underlying implementation and architecture
- Aaron Grattafiori, *Understanding and Hardening Linux Containers*, DEF CON 23
 - Kernel capabilities and advice for low-level security
- Docker documentation
 - Current state: a lack of “default on”
- Anthony Bettini, *Vulnerability Exploitation in Docker Containers*, Black Hat Europe 2015
 - Platform vulnerabilities
- Michael Cherney and Sagie Dulce, *Well, That Escalated Quickly! How Abusing Docker API Led to Remote Code Execution, Same Origin Bypass and Persistence in The Hypervisor via Shadow Containers*, Black Hat USA 2017
 - Targeting developers



Containerization & Docker

- Operating-system-level virtualization
- As an attacker, you're almost certainly already aware of hardware/platform virtualization
- “Lighter” virtualization
 - Shared kernel
 - Multiple user-space
 - Filesystems
 - Libraries
 - Networks
- Docker – Images, Containers, high level composition into applications
 - Development
 - Deployment



Vulnerabilities & Layers of Abstraction

- Vulnerability Life Cycle
 - Doesn't begin with discovery
 - Begins with a *mistake*
- *Everything* is an abstraction on top of physical properties of silicon
- Vulnerabilities are often a result of not understanding the layer(s) underneath you. Examples:
 - Web application vulnerabilities
 - Memory corruption and memory models
- ...or the “magic box” itself is broken

User Experience

Scripting Languages & OS services

High-level languages and OS APIs

Machine code and Virtual Memory

The Magic Box

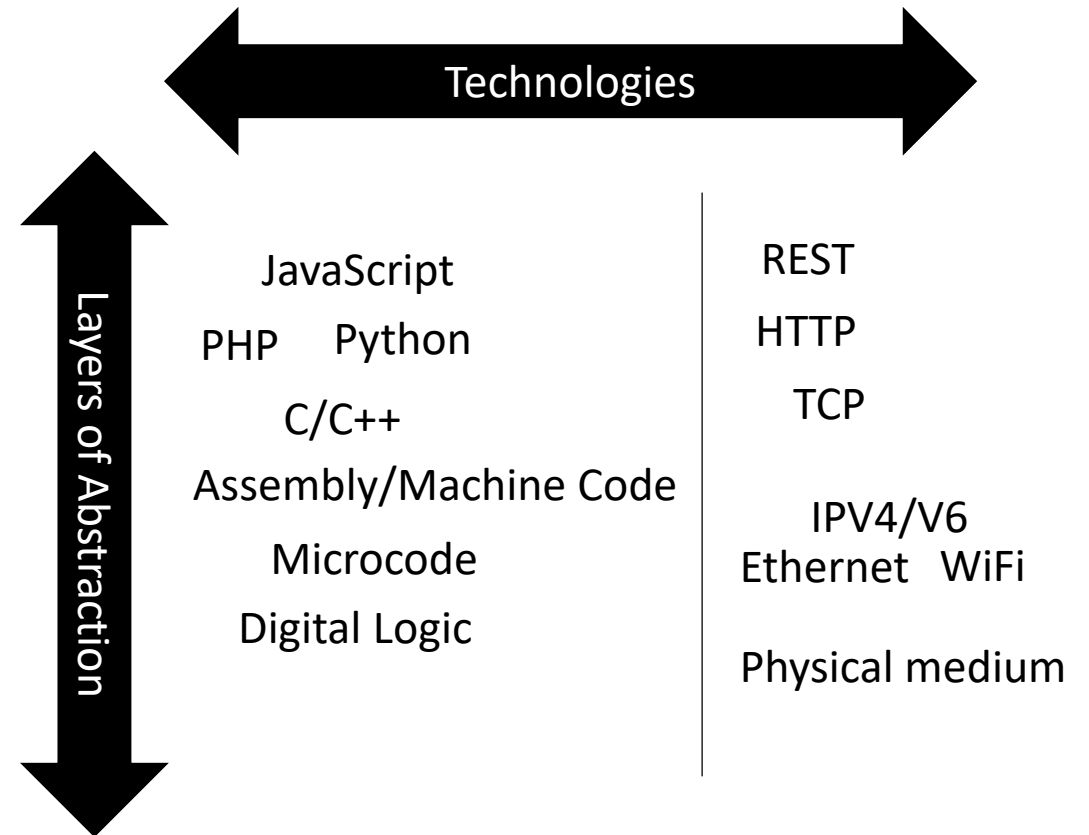
How Does a Hacker Keep Up?

- For your target, you don't get to dictate the attack surface and underlying environment:

- Language
- Protocols
- Platforms & Frameworks

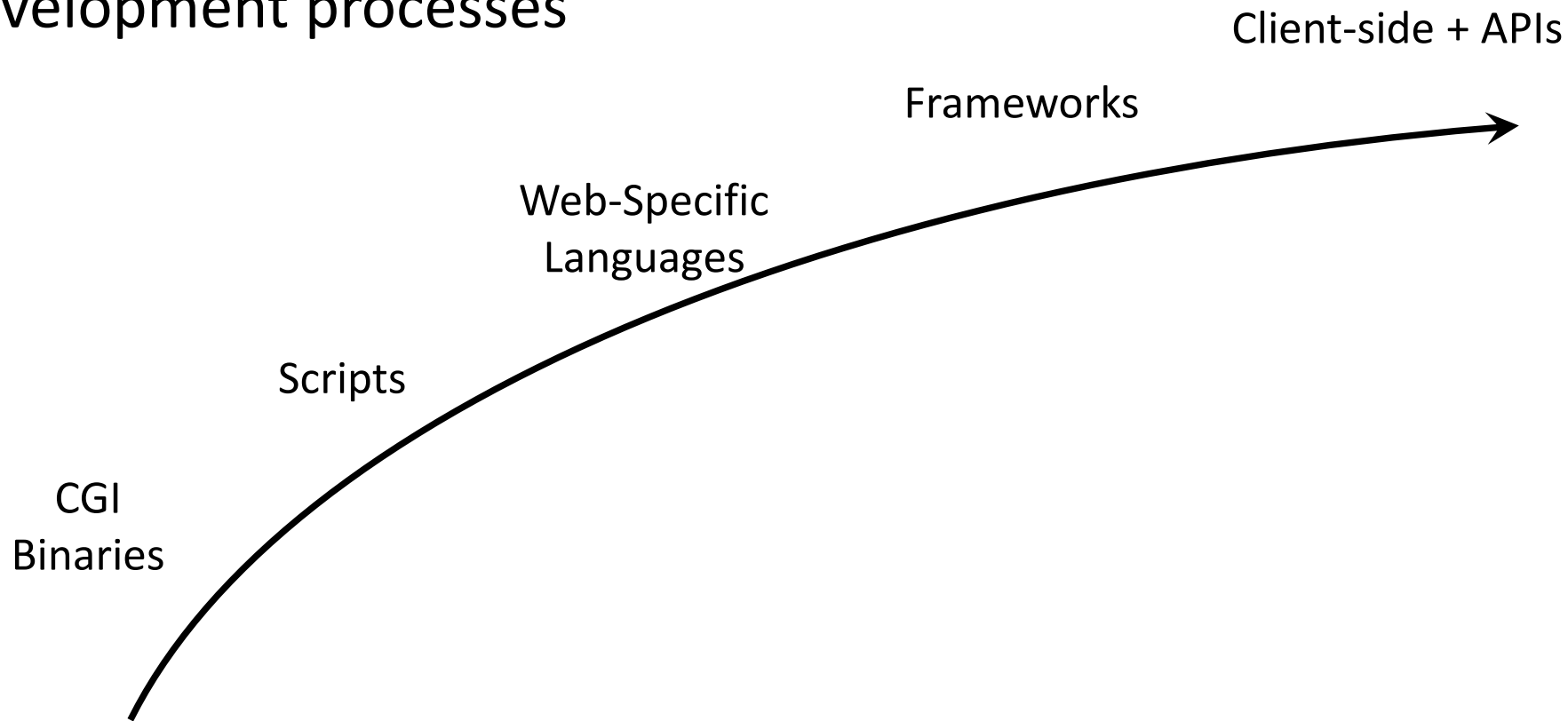
- Two dimensions of gap in skills

- Layers of abstraction
- Specifics of technology (above)

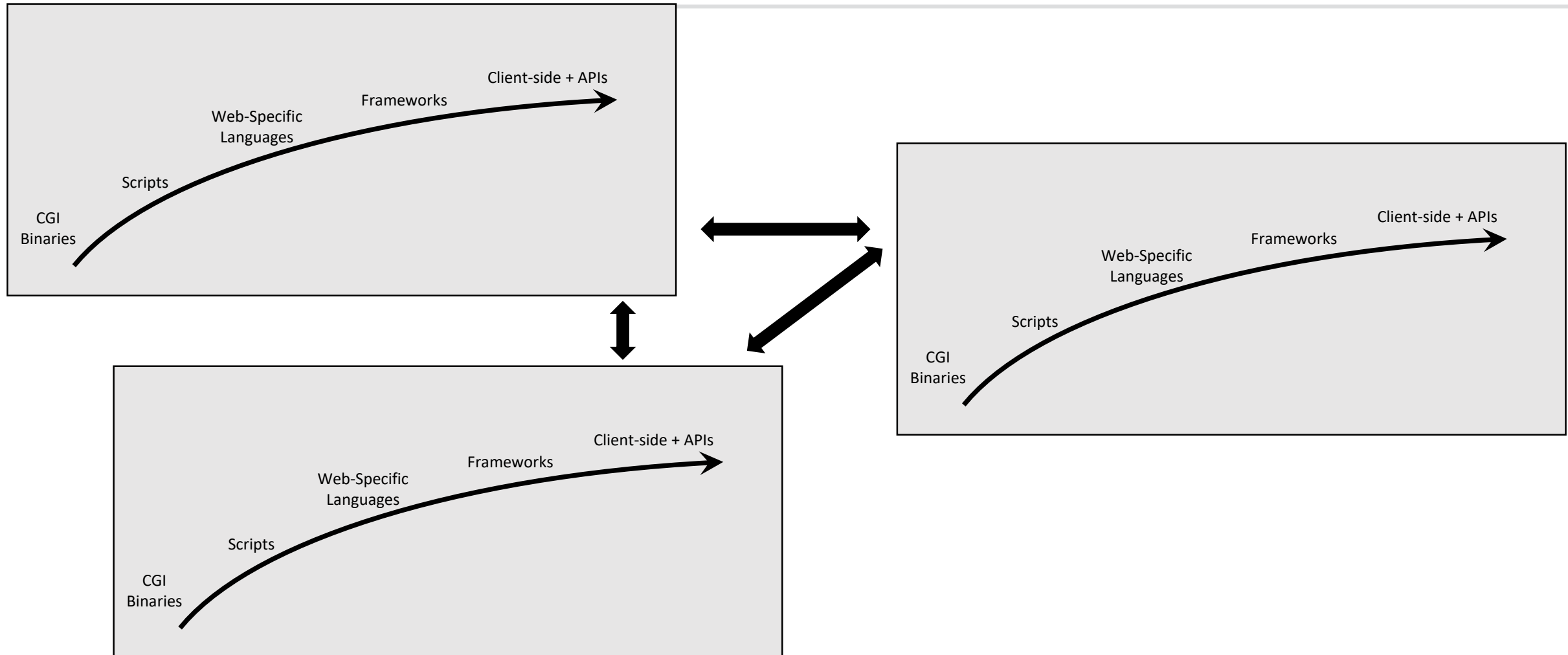


Movement and Abstraction in Development

- From lowest to highest-level abstraction in web application development processes



Next-Level Abstraction - Containerization





Mindset after you learn “Hello World”...

What can I build with these language constructs?

VS

How does “Hello World” work?



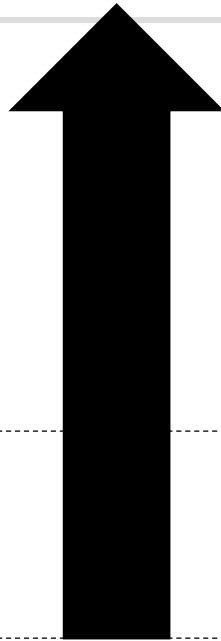
Movement and Abstraction for Hackers

- As an individual or small team you'll approach and become familiar with an increasingly large number of software projects
 - ...with more developers than an average CNE/CNA team
- Abstraction allows for more efficient development
 - Higher level technologies
 - Layers that “take care” of things for you
 - Lower-prerequisites for developers
 - Building block containers of mixed-technology software combined to make an application VS writing it in a monolithic style at a lower level
- What does this mean?

Movement and Abstraction for Developers vs. Hackers

Abstraction

Average
Developer
Work



Less-than-good news: Needs purposeful development/training
Becomes more important
New bug classes

Good news: Becomes more lucrative over time

Natural – “How does *X* work?”
More ‘interesting’ bugs

Typical Hacker
Skill
Development



Application to Attacking Application Internals

- Control over execution – opportunity to turn code against itself
 - Ex. – Malware analysis, ROP, Web API's, CSRF
- Skillset – Penetration tester vs. Application security expert
 - External vs. internal application attack surface
- Penetration tests less-often involve new “creative” control over execution in monolithic binary applications
- Pentester training gap
 - Basic understanding of memory corruption – introductory and conceptual
 - Targeted on understanding tool use
 - Not sufficient to target modern applications/environments
 - vs. motivated, funded, organized attackers that have developed talent

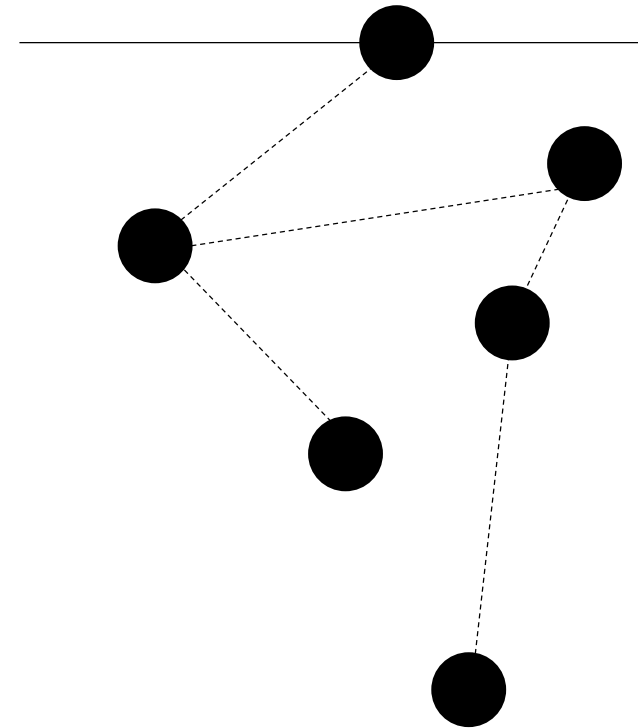


A Useful Shift for Attackers

- Containerization allows for the design of applications that are composed of many independent single-purpose services.
- Democratizing post-exploitation manipulation and instrumentation
 - Observing and instrumenting program flow/data
 - Monolithic - Language/platform-specific knowledge/tools
 - Multi-container – Leveraging system/network-level post-exploitation and sniffing tools

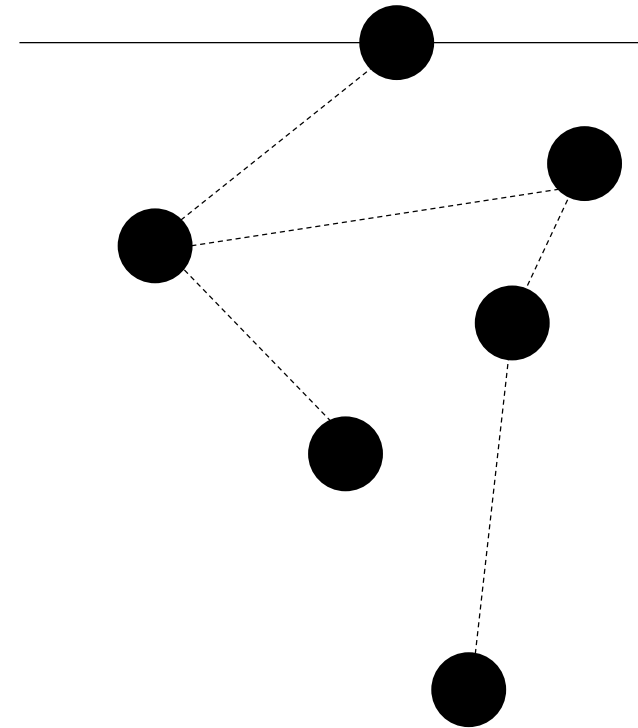
Taking Advantage of Abstraction

- Organization-wide attack
 - Progression/change of compromising connected systems
- Multi-container applications have their own networks (possibly shared with other applications)
 - The test of an application becomes a microcosm of an organization-wide test
 - The same would happen with traditional virtualization, but it'll be more common with containers: light and easy



Taking Advantage of Abstraction


- Exploitation of multi-container attack surface will begin with specific software in one container
- Post-initial-exploitation, access to an internal network of the *rest* of the containers, services, data, and protocols.
- Leveraged by the usual tools you already have familiarity with as a penetration tester.
- Analogous to hooking/examination/instrumentation of monolithic application





Docker as a Target Application Platform

- Monolithic container applications
 - Ease of deployment
- Multi-container applications
- Docker container networks
 - Default shared between containers
 - Configurations define which ports are “published” or shared to the outside world through the host.
 - Inside the network, containers may freely scan/connect/probe



Basic Exploration of Docker Container Applications

Quick connectivity check from one container to another, without the target container being explicitly configured to allow the connection



Implications

- Access through conventional exploits place attackers into an internal network with opportunity to pivot
- Familiar territory for attackers with system/network-level attack experience
- Limits: “Living off the land” is more challenging due to minimalistic images
- Learn to identify – You may not realize you’re inside of a Docker container network until you’ve exploited the external attack surface of it.



Exploitation and Post-Exploitation of a Multi-Container Application

Externally, with Kali & Metasploit:

Leveraging an older Joomla in the Docker Hub repositories

Pivoting to manipulation of multi-container voting application (from Docker tutorials)



Take-Aways

- Existing offense skills become useful at a lower relative position of abstraction relative to newer applications.
 - Developers are moving up, the new “low level” moves up
- Important to update yourself. Work “up” the stack as well.
 - New development practices such as multi-container application composition
 - Chase a trendy technology and look at the attack surface
- Containerization represents an opportunity for attackers to leverage existing network/system-level knowledge to explore the internals of applications that are composed of multiple containers.
 - Your existing skills are moving “down” the stack relative to where applications are being developed, and you can take advantage of it.



Discussion & Contact Information

Whitepaper available in conference materials, with more references and resources.

Wesley McGrew
Director of Cyber Operations
HORNE Cyber

wesley.mcgrew@hornecyber.com
[@mcgrewsecurity](#)