

{deepfactor}

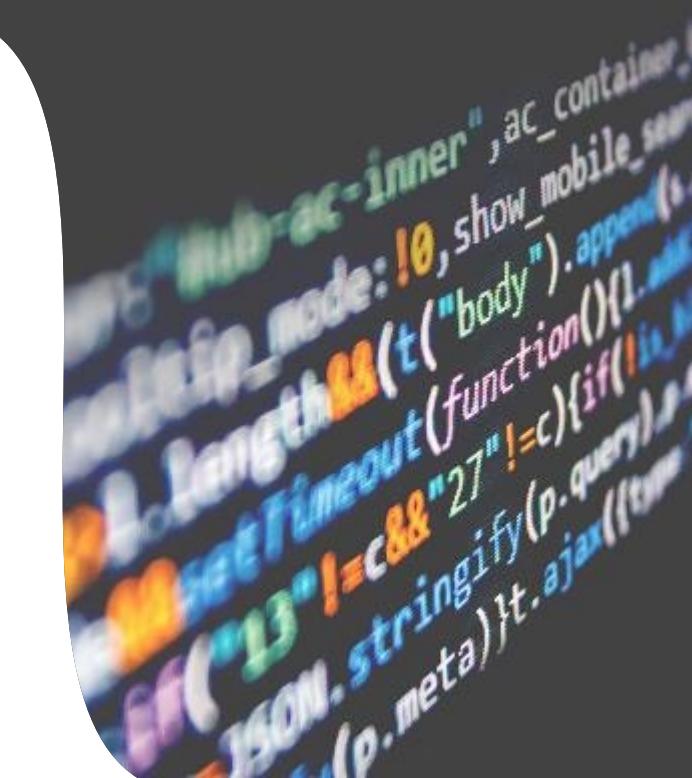


AppSec 2.0: Reimagine AppSec with Runtime Analysis

OWASP LA Meetup

August 23, 2023

KIRAN KAMITY, CEO AND FOUNDER, DEEPFACTOR



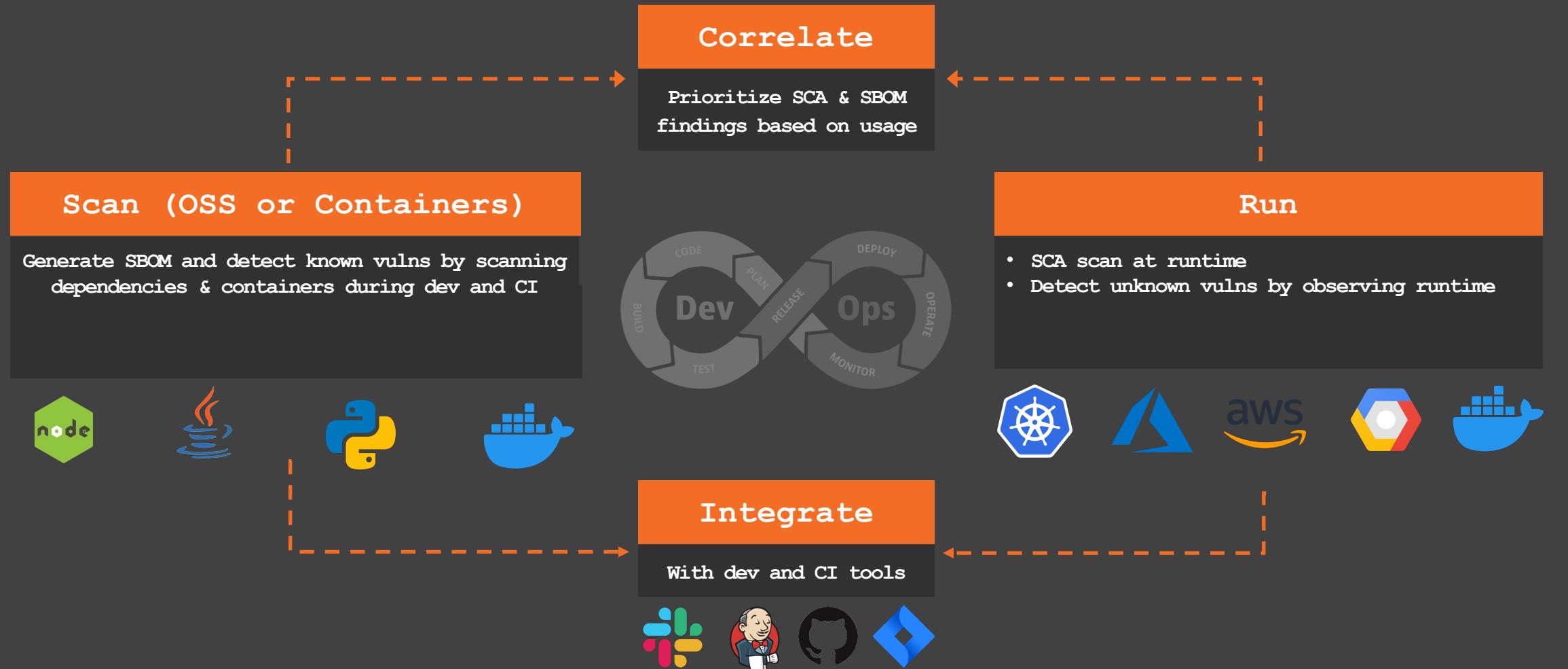
What we'll cover today:

1. Limitations of first-gen AppSec tools
2. AppSec 2.0 powered by runtime
3. What can it help us accomplish?
 - A. Runtime Enriched SCA
 - B. Runtime Security
4. Technical deep dive & Demo
#

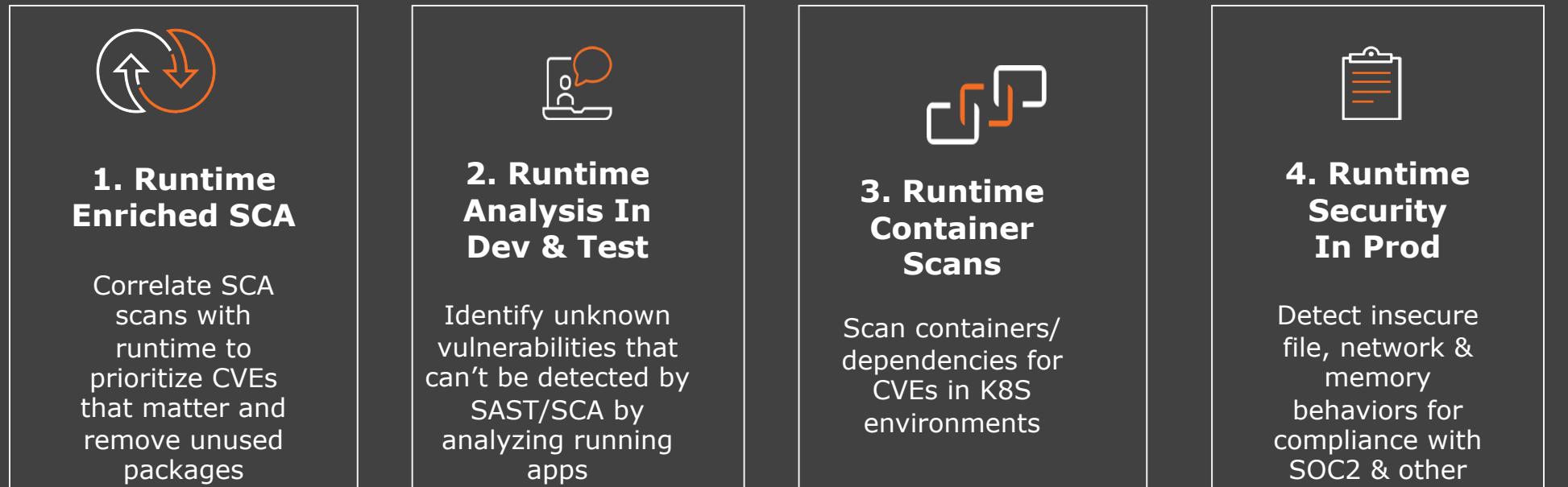
Alert Fatigue w/ SCA?



// AppSec 2.0



// How Can Runtime Visibility Turbocharge AppSec

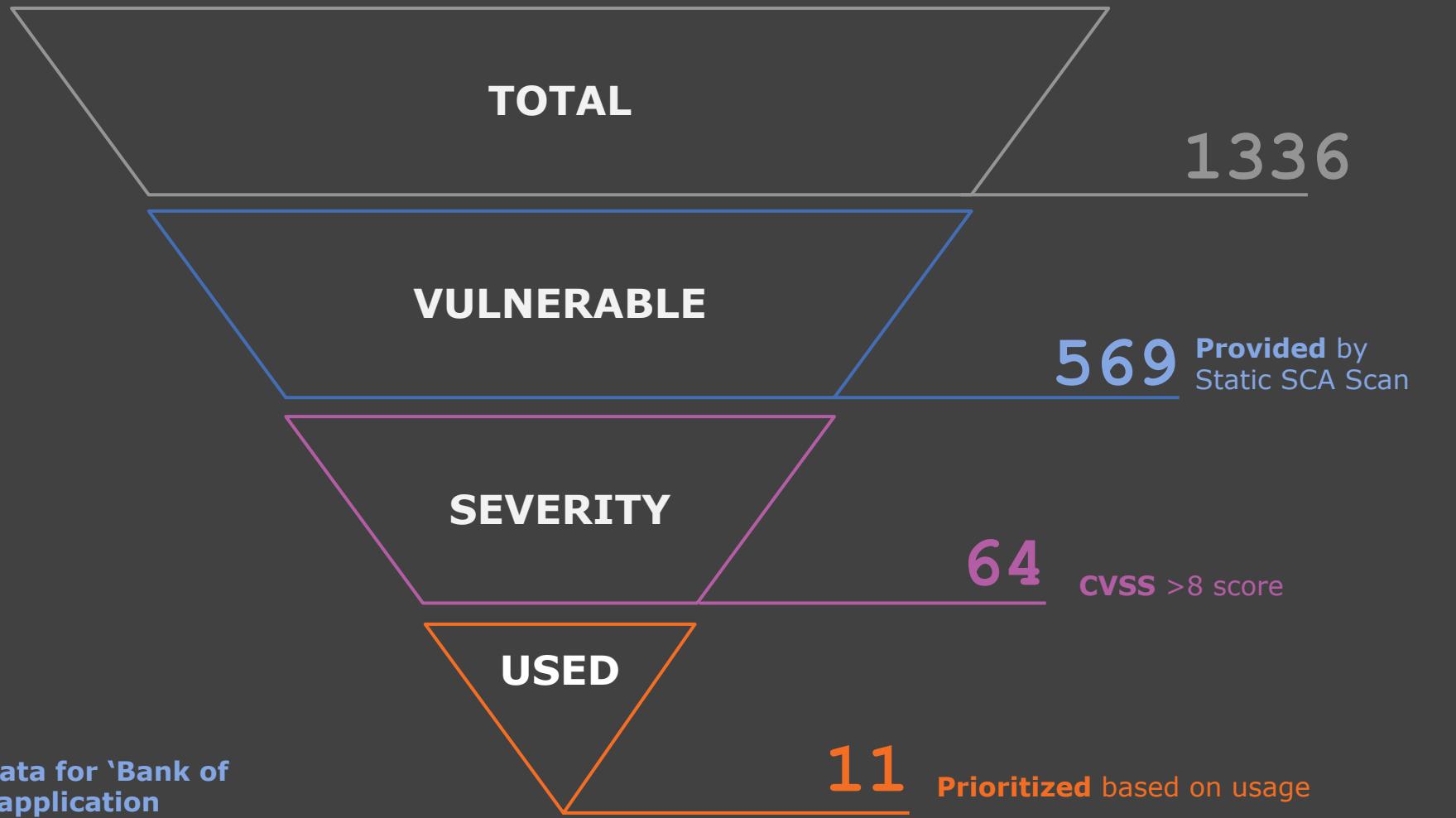


Runtime Software Composition Analysis

#

{df}

// The Holy Grail: Prioritize Based On Actual Usage



// Filtering SCA False Positives & Reducing Alerts



// Reachability & Usage: Technical Approaches

	Runtime Analysis	Call Graph Analysis
Ability to detect reachable modules in dependencies		
Support for OS packages used in containers		
Ability to detect actual usage for dependencies and/or OS packages		
Ability to detect dynamically loaded modules (such as reflections)		
De-prioritization of deadcode/unused code		
Language Agnostic		

Runtime Analysis & Security

#

{df}

// Detecting Insecure Behaviors at Runtime



- Execution
- File system behavior
- System calls
- Remote code execution
- Library behavior
- Memory behavior
- Network behavior
- Privilege escalation
- Buffer overflow
- Secrets

The screenshot shows the Deepfactor platform's alert policies interface. The left sidebar has icons for Home, Applications, API, and Settings. The main area shows the title "Alert Policies / Deepfactor Demo Policy". Below it, there are tabs for "Application Risks" (which is selected) and "OS Packages". The "Application Risks" tab lists several categories: Insecure Execution, Insecure File System Behavior, Improper Use Of System Calls, Remote Code Execution, Insecure Library Behavior, Insecure Memory Behavior, and Insecure Network Behavior. The "Privilege Escalation" category is highlighted with a pink background. To the right, under "Rules for Privilege Escalation", there are six alert rules, each with a severity level (P1 or P4), a description, and a "View Details" button. Each rule also includes an "Ignore if the following processes trigger this alert rule" section with a link to a process list.

Rule Type	Severity	Description	Action
Process	P1	Alert if process UID changes unexpectedly	View Details
Process	P1	Alert if process GID changes unexpectedly	View Details
Process	P4	Alert on use of setuid	View Details
Process	P4	Alert on use of setgid	View Details
Process	P4	Alert on use of setfsuid	View Details
Process	P4	Alert on use of setreuid/setgroups	View Details
Process	P4	Alert if application changes LD_LIBRARY_PATH after startup	View Details

Technical Deep-dive & Demo

#

{df}

// Userspace Interception vs eBPF

	Deepfactor	eBPF
Best suited for	Dev (shift left runtime security & correlation with SCA)	Ops (Live threat detection)
Technical Approach	LD_PRELOAD & language agents	Enabling eBPF in the host OS
Granularity	Enable at a process/container level	Enable at host level
Performance Overhead	Low	Low
Language Agents	Visibility into classes loaded etc. (required for correlation with SCA)	No visibility at the language agent level
Support for AWS Fargate, Lambda w/ containers, AWS ECS, Docker Swarm etc..	Supported (Docker build w/ libdf.so)	No support
Visibility into userspace of process	Yes (visibility into certificates, memory allocation behaviors, process behaviors ..)	No visibility above system call layer

{df}

// Observing Running Apps For Various Deployment Types

- Monolithic applications
 - dfctl run -a "my application" --cmd COMMAND_WITH_ARGUMENTS
- Containers
 - dfctl run -a "my application" --docker-run DOCKER_RUN_OPTIONS --image IMAGE
- Kubernetes
 - helm --install df-webhook-stable -n df-webhook deepfactor/webhook --set clusterName=CLUSTER_NAME_OF_YOUR_CHOICE --create-namespace -f webhook-override.yaml
- Docker Swarm, AWS ECS, Lambda w/ container ...
 - docker build -f Dockerfile.df --build-arg "APP_IMAGE=\${APP_IMAGE}" --build-arg "APP_IMAGE_ID=`docker inspect \${APP_IMAGE} --format '{{.Id}}'`" --build-arg "DF_APP_NAME=my application" .

// Demo

The image shows a tablet displaying the deepfactor developer security interface. The interface features a dark background with orange and white text. At the top, it says '{deepfactor}' and '// Developer Security'. Below that, it says 'Develop Secure Cloud Native Apps'. On the left side, there are five cards with icons and descriptions:

- SBOM**: Generate SBOMs to secure your supply chain and comply with U.S. Executive order 14028
- SCA**: Scan containers/dependencies for CVEs based and gate builds and pull requests
- Runtime Enriched SCA**: Prioritize SCA findings based on correlation with runtime usage behavior
- Runtime Analysis**: Identify unknown vulnerabilities by analyzing running apps in Dev & Test
- Runtime Security**: Detect insecure file, network, and memory behavior in production environments

On the right side of the tablet, there is a 'Sign In' form with fields for 'Email Address' and 'Password', and a 'Sign In' button. At the bottom of the tablet screen, there is a small note: 'BY LOGGING IN OR SIGNING UP, YOU AGREE TO ABIDE BY OUR POLICIES, INCLUDING OUR TERMS OF SERVICE AND PRIVACY POLICY.'

{df}

// Learn More?

Deepfactor Free Trial



An integrated approach to next-gen AppSec:
SCA, SBOM, Container Scans, Runtime
SCA & Runtime Security

{df}

SCA 2.0 Whitepaper



A Framework to Prioritize Risk, Reduce
False Positives, and Eliminate SCA Alert
Fatigue”

Questions?

#

{df}

```
"LIMIT_total:" + d);
: " + d)); var n =
[g]), -1 < e && b.sp
", word:c[g]}});
);
e = m(b, "");
push(b[c].b). "param
```

```
ect(a, b), a = new user(a),
= 0;c < a.length;c++) {      use_ar
or (var b = "", c = 0;c < a.length;c+
AttrModified textInput input change ke,
ords + " UNIQUE: " + a.unique); $(#inp
e().unique);}); function curr_input_unique
a.length) { return ""; } for (var i =
",); if (i < a.length;c++) {
$( "#User_logged").val(b);
c < a.length;c++) {
length - 1; return
a[c], b) && b.push(a[c]);
).val(), b = b.replace(/\s+/g, " ");
_array = b.split(" ");
gth;a++) { 0 == u
b[b.length - 1].use_a
a.sort(dynamicSort("u
), b = indexOf_keyword(
); b = indexOf_keyword(
).splice(b, 1); return a;
se_array(a, b) { for (va
z_array(a, b) { for (va
word(a, b) { fo
} } return c;
ction(c, d) {
"; b += "; i
} { if (f = a.
if ("#go-but
```

{deepfactor}

Thank you

Kiran Kamity

Connect on [Linkedin](#)

Email: kiran@deepfactor.io