PhD Dissertation & Backgrounds

VIRGINIA TECH
DEP. OF COMPUTER SCIENCE
PHD (2021.5)

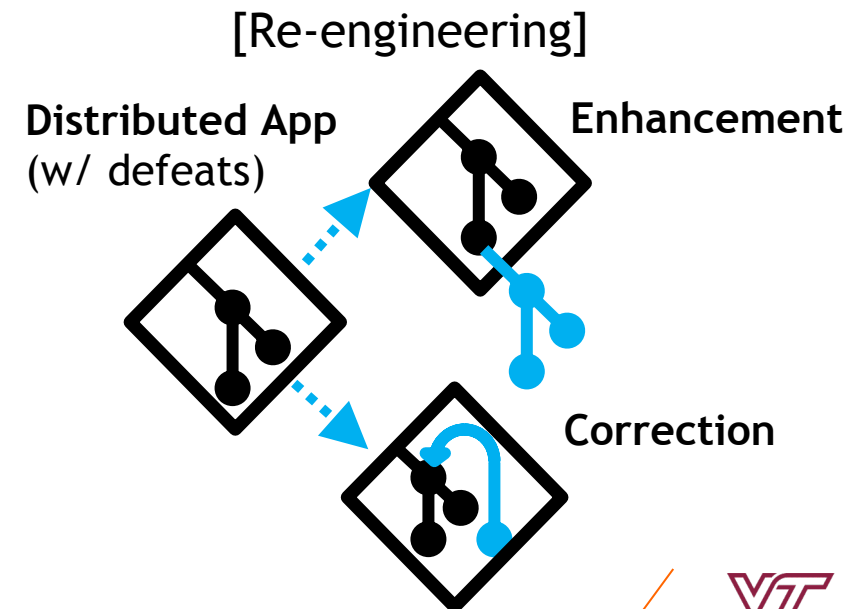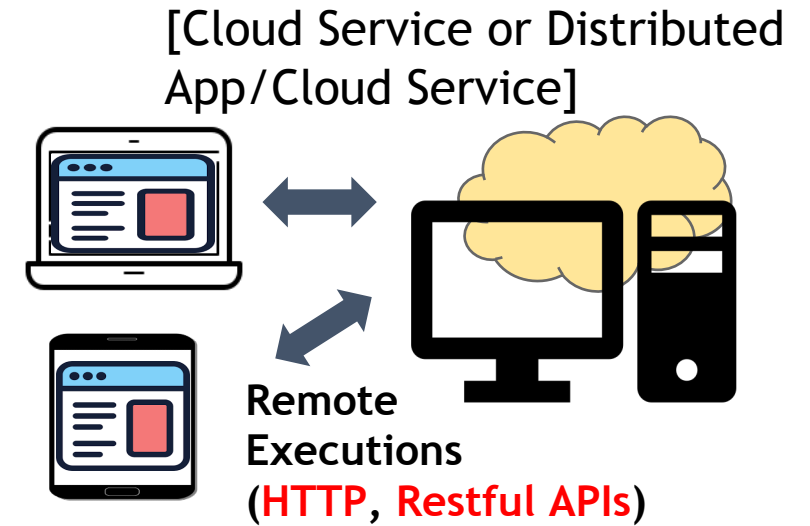## KIJIN AN

https://kjproj84.github.io

Dissertation Contributions

- Advisor: **Eli Tilevich**

- My Research Topic:

  - Software Engineering,

  - Distributed Systems (Web),

  - Computer Networking

Dr. Tilevich   Dr. Smaragdakis

- New "**Refactoring**": "Client Insourcing"

  - *Creating a <u>Centralized Variant</u> ($\mathcal{P}'$) for the Distributed App/Cloud Service ($\mathcal{P}$)*

- Value and Utility of "Client Insourcing"

  - *"Pinpointing"* Inefficiency of Distributed Programs and *"Assisting"* Programmers for their changes

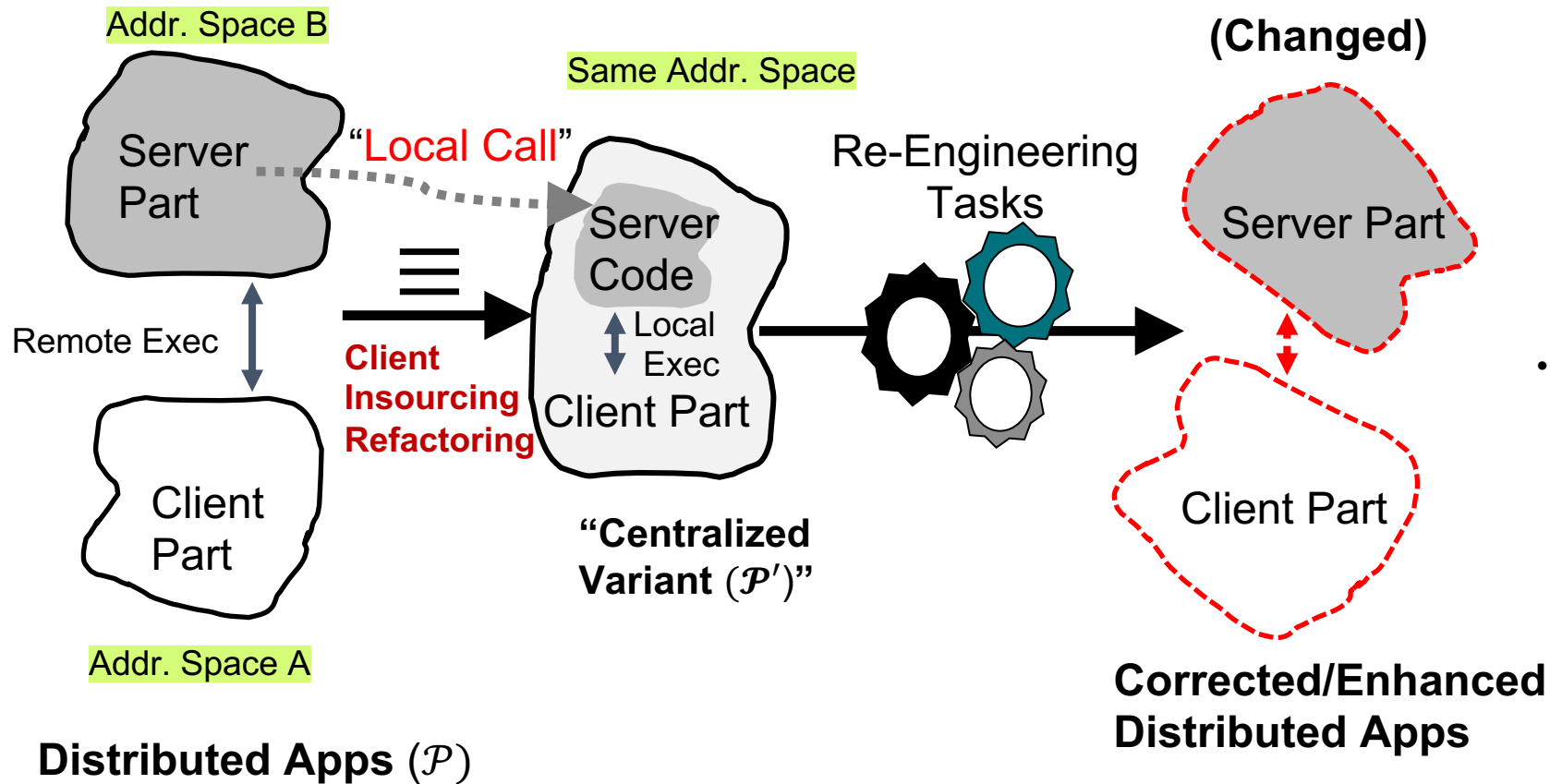 - Applying state-of-the-art techniques from **Software Engineering** to address problems in Distributed Apps
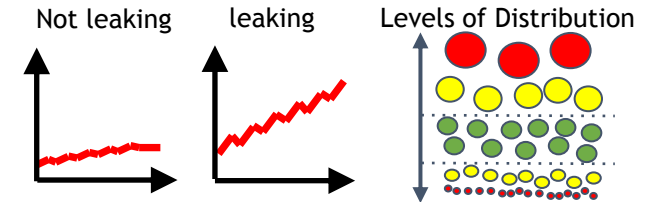
[Cloud Service or Distributed App/Cloud Service]

Remote Executions (**HTTP**, **Restful APIs**)

[Re-engineering]

Distributed App (w/ defeats)

Enhancement

Correction

VIRGINIA TECH.

Client Insourcing Refactoring

Addr. Space B

Server Part

"Local Call"

Remote Exec

Client Part

Addr. Space A

Distributed Apps ($\mathcal{P}$)

Same Addr. Space

Client Insourcing Refactoring

Server Code

Local Exec

Client Part

"Centralized Variant ($\mathcal{P}'$)**"**

Re-Engineering Tasks

(Changed)

Server Part

Client Part

Corrected/Enhanced Distributed Apps

Correcting Distributed Apps

Debugging <u>Memory</u> Leakage/<u>Performance</u> Bottlenecks (very quickly)

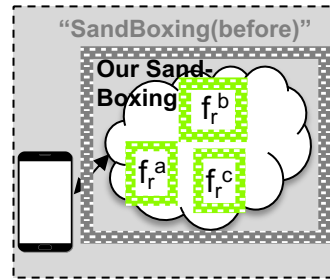Optimizing <u>Distribution</u> <u>Granuality</u>

Not leaking   leaking   Levels of Distribution

Enhancing/Adapting Distributed Apps

Sandboxing with minimum overheads

Replicating Distributed Apps for Mobiles/Edges

"SandBoxing(before)"

Our Sand-Boxing

f_r^b

f_r^a f_r^c

Replica_1

Cloud App

Replica_2

Original   Replica_k   …

VIRGINIA TECH

Client Insourcing Refactoring [WWW 2020]

```
//CLIENT: app/../property-details.ts
unfavorite(event, property){
                        //Marshalling
 this.pServ.unfavorite(property)
 .subscribe(favorite //unMarshalling
     =>{ this.favorites = favorites};
}
```

1) HTTP **Request** From Client

```
DELETE
/properties/favorite
HOST ..
[{"id":1,"city":..}]
```
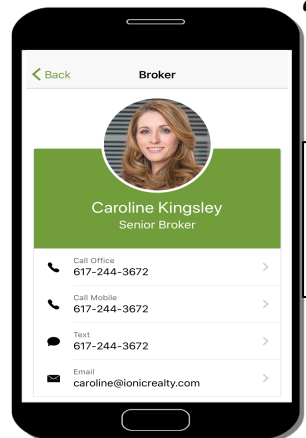
Entry Point

2) HTTP **Response** From Server

```
HTTP/1.1 200 OK
Content-t: json
Content-Len: …
[{"id":2,"city"…]
```

Exit Point

```
//SERVER:server.js
app.delete('/properties/favorites/'
, properties.unfavorite);
//server/properties.js
var favorites = require('./property').favs;
function unfavorite(request, response) {
 var id = request.body.id;//unMarshalling
  for (var i=0; i<favorites.length; i++){
   if (favorites[i].id == id){
     favorites.splice(i, 1);
     break;}}
  response.json( favorites )//Marshalling
}
//SERVER: server/property.js
exports.data = [{id: 1,...}];
exports.favs = [{id:2,…},…,{…}];
```
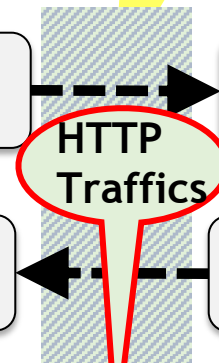
(Correspondence for "*Marshaling*" Points)

Caroline Kingsley
Senior Broker

Call Office
617-244-3672

Call Mobile
617-244-3672

Text
617-244-3672

Email
caroline@ionicrealty.com

Client Code

Marshaling **client-param**

Unmarshaling **server-return**

Send REQ

Get RES

HTTP Traffics

Get REQ

Send RES

Unmarshaling **client-param**

Marshaling **server-return**

Server Code

(Client-side)

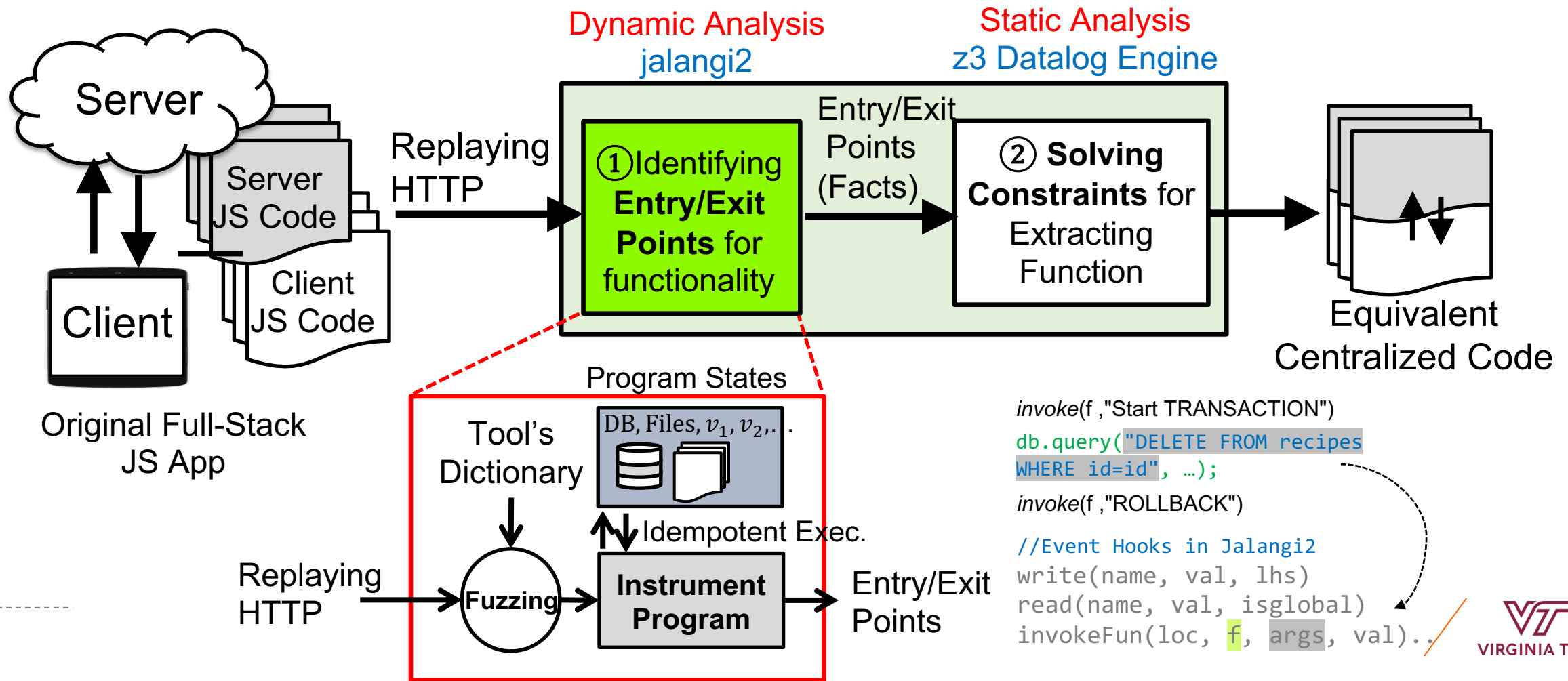(Server-side)

Replaying/Capturing

1. Decode {**client-parameter**, **server-return**}
2. Instrument code parts that RWs these values

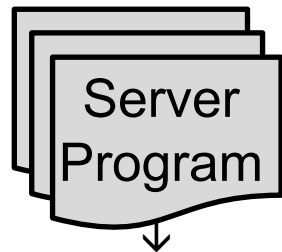VIRGINIA TECH

Client Insourcing Refactoring [WWW 2020]

■ **Fuzzing HTTP** records**, Idempotent** Executions

Dynamic Analysis
jalangi2

Static Analysis
z3 Datalog Engine

Server

Server JS Code

Client

Client JS Code

Replaying HTTP

① Identifying **Entry/Exit Points** for functionality

Entry/Exit Points (Facts)

② **Solving Constraints** for Extracting Function

Equivalent Centralized Code

Original Full-Stack JS App

Program States

DB, Files, $v_1, v_2, ...$

Tool's Dictionary

Idempotent Exec.

Replaying HTTP

Fuzzing

Instrument Program

Entry/Exit Points

```
invoke(f ,"Start TRANSACTION")
db.query("DELETE FROM recipes
WHERE id=id", …);
invoke(f ,"ROLLBACK")

//Event Hooks in Jalangi2
write(name, val, lhs)
read(name, val, isglobal)
invokeFun(loc, f, args, val)..
```

5

Client Insourcing Refactoring [WWW 2020]

- Extracting Function: Searching all dependent JS code in Entry/Exit points

- Extending *Declarative approach for Program* Analysis (z3py, Datalog Engine)

  - JavaScript**: GATEKEEPER** [Security'09]: Point-to-Analysis, **JSDep** [FSE'15]: Dependency Analysis

;;Generating Facts

;;Rules for Client Insourcing Refactoring

;;z3 Datalog Engine
Query ExectedStmts for specific HTTP method
{Client Param, Server Return}

Server Program

$\mathbf{DataDep}(s_1, \mathrm{stmt}_2) \leftarrow \mathbf{Read}(s_1, v_1) \wedge \mathbf{Write}(s_2, v_1)$

;;JS-Dep, GATEKEEPER

$\mathbf{UnMar}(s_1, v_{unMar}, V_{unMar}^{u_{id}})$
$\quad \leftarrow \mathbf{Write}(s_1, v_{unMar}) \wedge \mathbf{Ref}(v_{unMar}, V_{unMar}^{u_{id}})$

$\mathbf{Marshal}(s_1, v_{Mar}, V_{Mar}^{u_{id}})$
$\quad \leftarrow \mathbf{Write}(s_1, v_{Mar}) \wedge \mathbf{Ref}(v_{Mar}, V_{Mar}^{u_{id}})$

$\mathbf{ExecutedStmts}(s_n, V_{unMar}^{u_{id}}, V_{Mar}^{u_{id}})$
$\quad \leftarrow \left(\mathbf{DataDep}(s_n, s_1) \wedge \mathbf{Marshal}(s_1, v_{Mar}, V_{Mar}^{u_{id}})\right) \wedge$
$\quad \left(\neg\mathbf{DataDep}(s_n, s_2) \wedge \mathbf{UnMar}(s_1, v_{unMar}, V_{unMar}^{u_{id}})\right)$

$\mathbf{Write}(s_1, v\_a)$
$\mathbf{Read}(s_2, v\_b)$
$\mathbf{Write}(s_2, v\_b)$
...
$\mathbf{Ref}(s_c, v\_c, V_1)$
$\mathbf{Ref}(s_d, v\_d, V_2)$

Original Server Code

"*Extract Function*" Refactoring

Transforming Client Code

Resulting Centralized Program
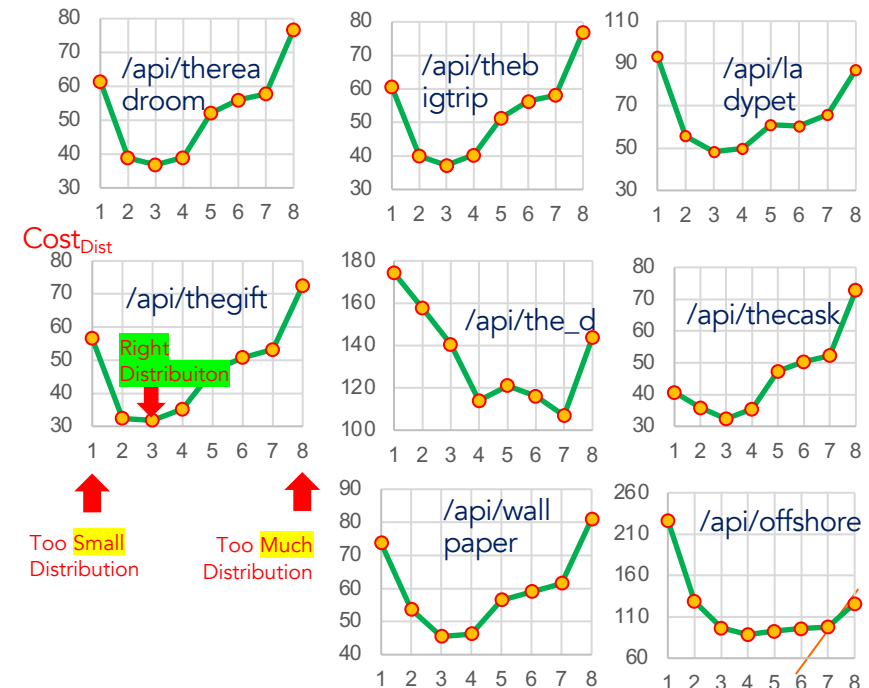
Original Client Code

6

Application 1: Optimizing Cloud Services [SANER 2020]
Restructuring Distribution

- Correcting ill-conceived Distributions
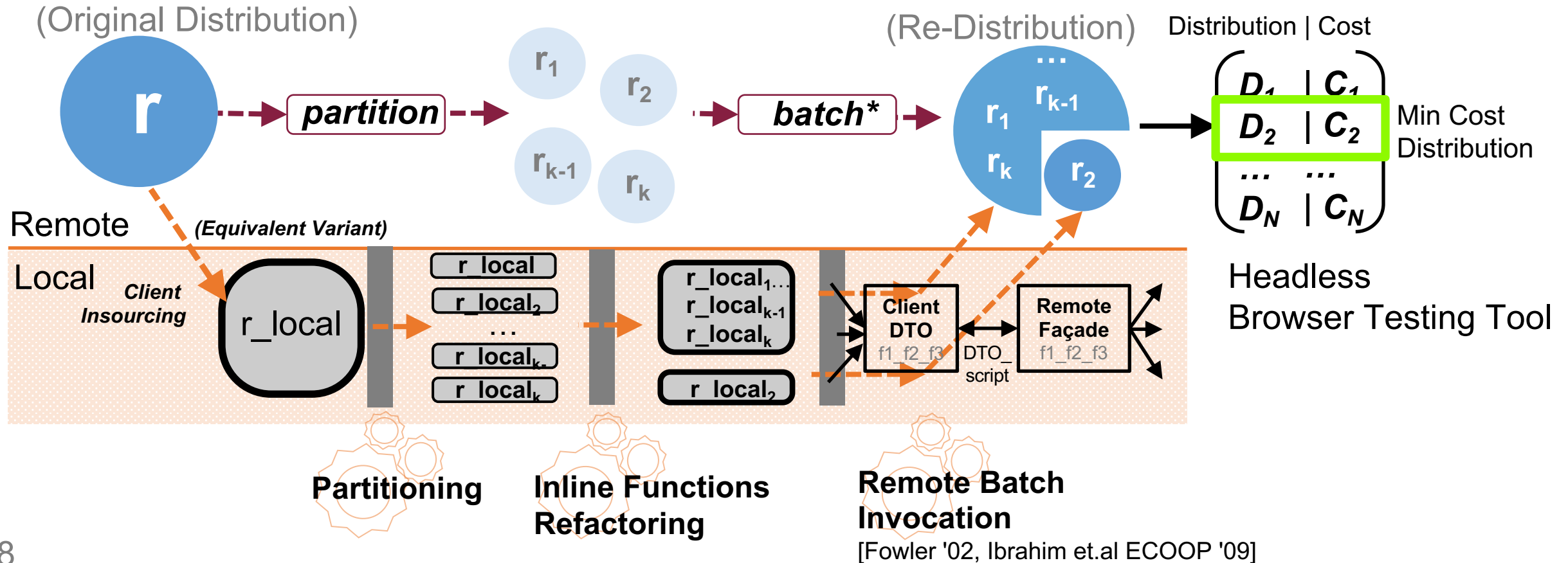 - Ex) Nano-service anti pattern

Too Small Distribution

getVoc GetC
GetSenGetDial
getCol

(Right boundary)

"**Right**" Redistribution

getVoc

GetC GetDial

GetSen getCol

Level of granularity

Too Much Distribution

getVoc   GetC

Commutation Overheads

GetSen   GetDial

getCol

Client-Side   **Server-Side**
(`/api/ladydog`)

- Determine which functional distribution would minimize the cost of distributions

  - $C_{Dist\_Exec}(\mathbf{r}) = \alpha \cdot latency(\mathbf{r}) + (1-\alpha) \cdot \Sigma\ resource(\mathbf{r})$

- Large Distribution Space: Our Tool automates!

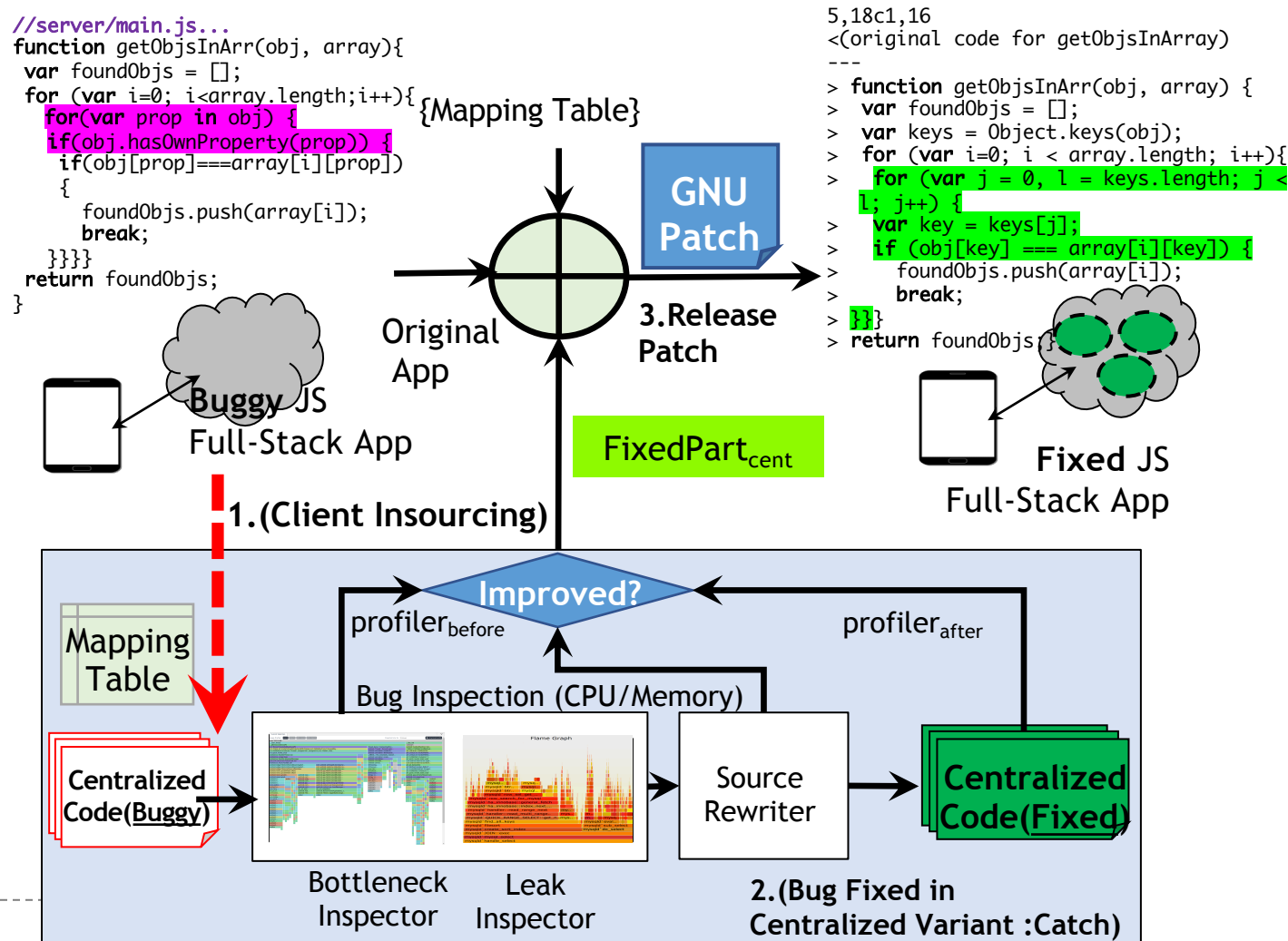 - Ex) $394 \times 4139 \sim= 1.6 \times 10^{6}$ ULOCs

/api/therea droom

/api/theb igtrip

/api/la dypet

$Cost_{Dist}$

/api/thegift

Right Distribuiton

/api/the_d

/api/thecask

Too Small Distribution

Too Much Distribution

/api/wall paper

/api/offshore

Application 1: Optimizing Cloud Services [SANER 2020]
Restructuring Distribution

(Original Distribution)

r

partition

r_1 r_2 r_{k-1} r_k

batch*

(Re-Distribution)

... r_{k-1} r_1 r_k r_2

Distribution | Cost

$$\begin{pmatrix} D_1 & | & C_1 \\ D_2 & | & C_2 \\ ... & | & ... \\ D_N & | & C_N \end{pmatrix}$$

Min Cost Distribution

Remote

Local

(Equivalent Variant)

Client Insourcing

r_local

r_local r_local$_2$ ... r_local$_{k-}$ r_local$_k$

r_local$_{1...}$ r_local$_{k-1}$ r_local$_k$ r_local$_2$

Client DTO f1_f2_f3

DTO_ script

Remote Façade f1_f2_f3

Headless Browser Testing Tool

Partitioning

Inline Functions Refactoring

Remote Batch Invocation

[Fowler '02, Ibrahim et.al ECOOP '09]

8

VIRGINIA TECH

```
//server/main.js...
function getObjsInArr(obj, array){
 var foundObjs = [];
 for (var i=0; i<array.length;i++){
   for(var prop in obj) {
   if(obj.hasOwnProperty(prop)) {
    if(obj[prop]===array[i][prop])
    {
      foundObjs.push(array[i]);
      break;
    }}}}
 return foundObjs;
}
```

{Mapping Table}

GNU Patch

3.Release Patch

Original App

Buggy JS Full-Stack App

FixedPart_cent

```
5,18c1,16
<(original code for getObjsInArray)
---
> function getObjsInArr(obj, array) {
>   var foundObjs = [];
>   var keys = Object.keys(obj);
>   for (var i=0; i < array.length; i++){
>     for (var j = 0, l = keys.length; j <
l; j++) {
>     var key = keys[j];
>     if (obj[key] === array[i][key]) {
>       foundObjs.push(array[i]);
>       break;
>   }}
> }}
> return foundObjs;}
```

Fixed JS Full-Stack App

1.(Client Insourcing)

Mapping Table

profiler_before   **Improved?**   profiler_after

Bug Inspection (CPU/Memory)

Centralized Code(**Buggy**)

Bottleneck Inspector   Leak Inspector

Source Rewriter

Centralized Code(Fixed)

2.(Bug Fixed in Centralized Variant :Catch)
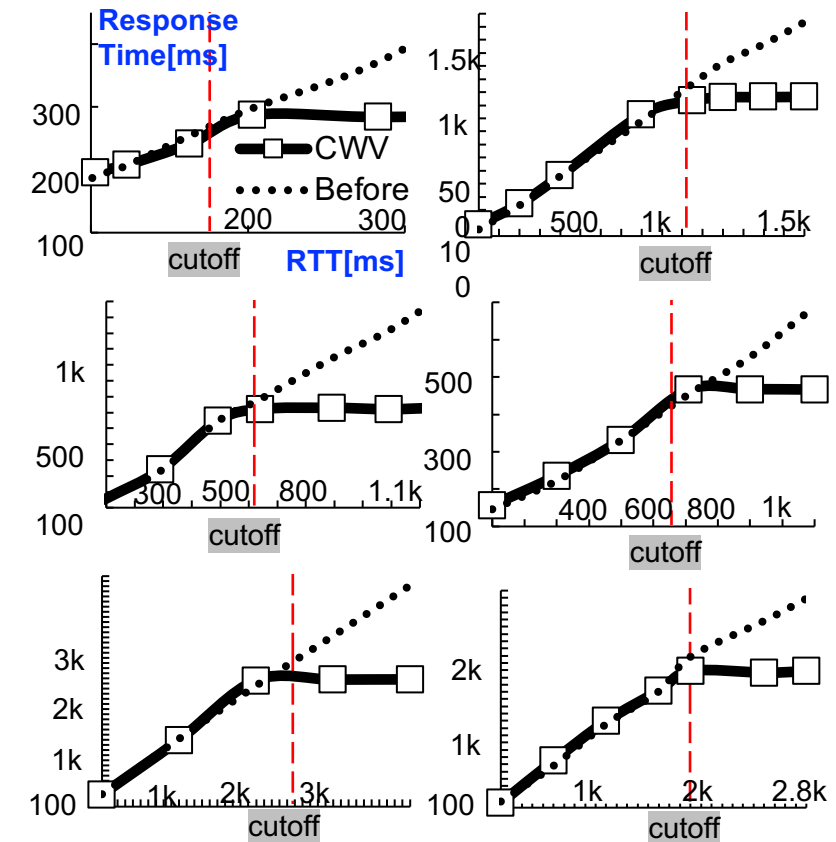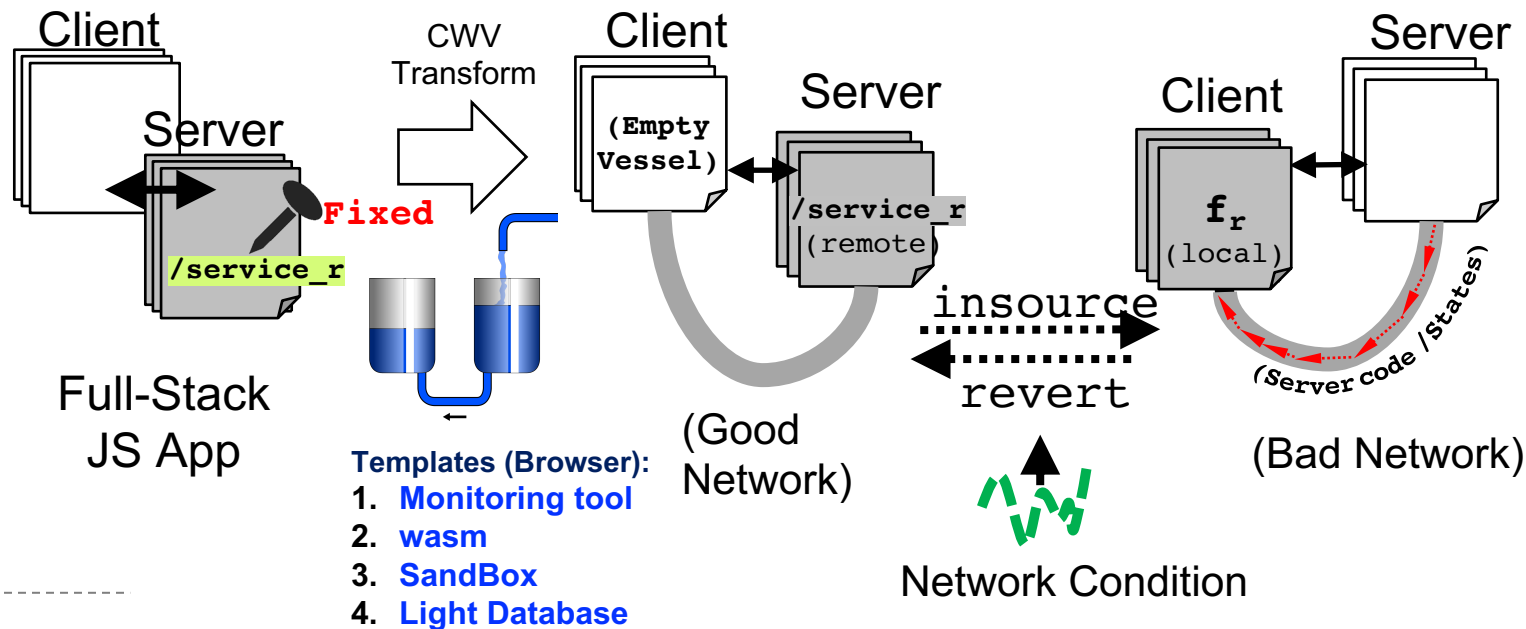
- Fixing Bugs in Centralized Variants and Generating Patches

- **90%** Reduced Time to execute Debugging Task

<Conventional Debugging>   <Our Simplification>

Reduced Steps

$$\frac{P_{before} - P_{after}}{P_{before}}$$

$\Delta T_{cent}(\%)$

$\Delta T_{dist}(\%)$

/users/search/id **Inefficient loop**

/users/search

/favorite

/api/big

/api/wall

/api/theg

/api/thered

/api/lady

/api/thed

/api/amont

misused APIs

/api/offs

VIRGINIA TECH

9

Application 3: Communicating Web Vessels (**CWV**)
Best Paper Award 🏆 [ICWE 2021]

- **Design** and **Execution Time** Mismatch – Client/Server Arch.

- Client *Insource* or *Revert* {**Code$_{Server}$**, **State**} based on Exec Conditions

- Automated Program Transformation for Adaptive Arch.

Publications & Honors in PhD

(full list: https://kjproj84.github.io/publications)

No.	Paper	Conference	Area	
1.	Client Insourcing Refactoring	**WWW 2020** (19%, 217/1129, Top-tier)	System (Web)	1st Author/2
2.	D-Goldilocks	**SANER 2020** (21%, 42/199)	Software Engineering	1st Author/2
3.	Catch&Release (Debugging)	**ICWE 2019** (25%, 26/106)	System (Web)	1st Author/2
4.	Comm Web Vessels	**ICWE 2021** (17%, 22/128, Best Paper 🏆)	System (Web)	1st Author/2
5.	EdgeFy: Edge-based framework	**Submitted**	System (Middleware)	1st Author/2
6.	**[Appendix]** **Project1**: Differencing Cross-platform Apps	**MobileSoft 2018** (Nominated for Best Paper)	Software Engineering	1st Author/3
7.	**[Appendix]** **Project2**: Distributing Embedded Apps for Trusted Exec.	**GPCE 2018**	Software Engineering	2nd Author/3
8.		**Journal of Com. Lang.** (Nominated for Best Paper)	Software Engineering	2nd Author/3

- Main work presented in WWW 2020 (Top-tier)

- One Best Paper Award & Two Best Paper Nominations

- Two Doctoral Symposium Papers in WWW 2020 and ICWE 2019

- Two Spotlights from CS@VT

VIRGINIA TECH

Before PhD Program: 1. "Computer Networking"
2. "Cloud-based Distributed Systems"

Sensor Units X *N*

(Networked)

Scale /Fault Tolerant

Distributed System

{Who/ Where/ Wat}

Our's Video PSNR

Delay Requirement For Video (buffering)

Unlikely to D_max

Route Maintenance

Request Route

Route Urgency_min

Drop Packets

Packet Scheduling

Most Urgent

Route Construction

Node Urgency

Service Next

D_max

(sharing paramters)

App Layer — D_max → **Network Layer** — |R|/U_node ↔ **MAC Layer**

VoCodec & Qualty   Call Tests (RTP/SIP)

[WiFi/VoIP System for Business Solution]

[WiFi/Repeater System for Metro Environment]

Donor — BTS / TDM / L2 Switch — Optical Lines — mAP#1~4 / MediaGateway (L2) / eRO — Monitoring WiFis / BiasT — eAP#1~6

2009 ——— **2012** — Industry #1 ——— **2015** — Industry #2

VIRGINIA TECH

Value Creating University
POSTECH
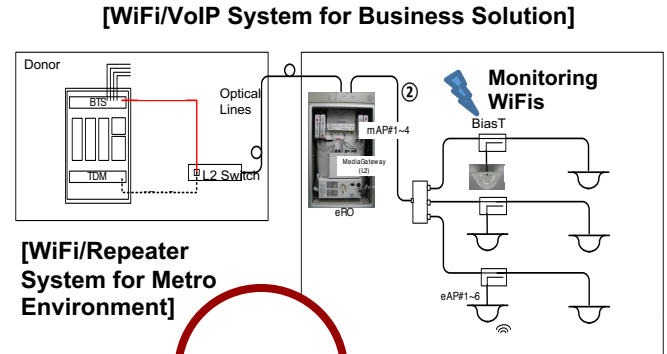POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

MS: Computer Networking (2 years)

- **MS Thesis: A Cross-layer** Scheme for **Video Data** Transmission
- **Routing** Protocols for Ad hoc Networking: AODV, DSR
- **MAC** Scheduler: WLAN or Zigbee
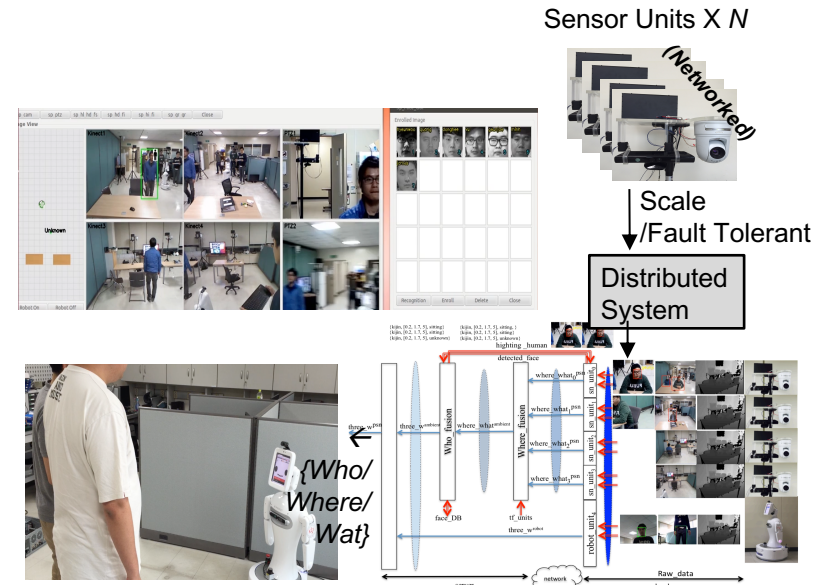- ICC 2009 (Conf), ACM/Springer Wireless Network 2013 (Journal)

SK telesys

(Network) System Software Engineer (3 years 4 months)

- Developing/Optimizing **Base Station**'s **MAC Protocols**
- WiFi/Repeater System : Remote Management tool for Metro (TR-069)
- Developing Business Functions for WiFi/VoiP System (Asterisk, SIP/RTP)

KIST 한국과학기술연구원
Korea Institute of Science and Technology

(Top Korean National Lab)
Software Engineer/Researcher (2 years 10 months)

- **Cloud-based Distributed System** for a Robot Service
- **Scale/Fault Tolerant** for **Sensor Units** x *N*
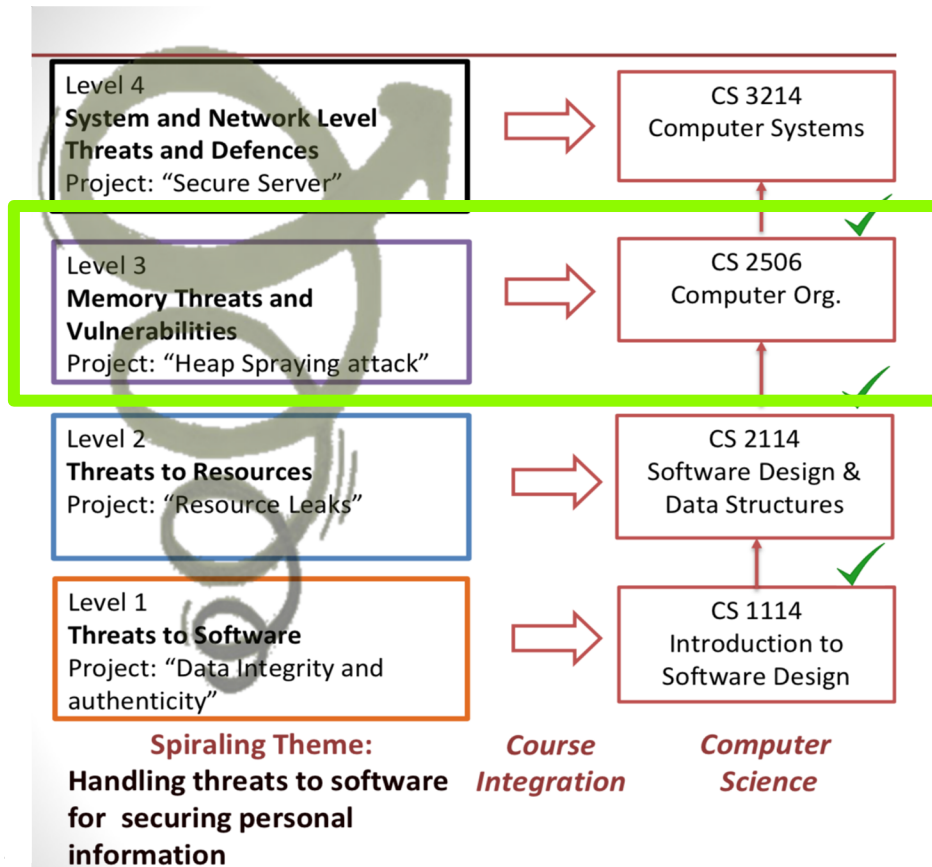- **Web-based Service Scheduler**

Beginning my PhD Program (2015.8~)

VIRGINIA TECH

Appendix

- ✓ Other Projects
- ✓ System Design Experience

VIRGINIA TECH.

Appendix: Understanding Heap Spraying Attacks

User Study (Pre/Post Survey): IRB, 540 undergrads for 3 Semesters (see more detail in https://kjproj84.github.io/teaching)

- NSF-funded project for increasing CyberSecurity-related education in CS and ECE core courses at Virginia Tech
- I developed the Level 3: *Understanding Heap Spraying Attacks*
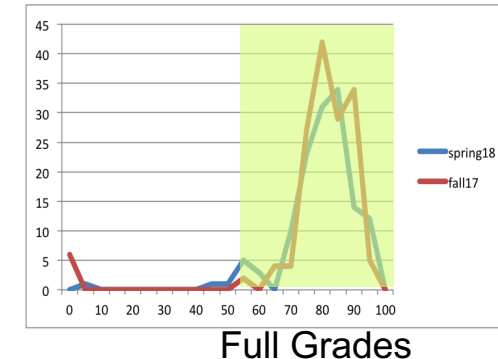

Very Good Feedback from Students

"This assignment was very challenging, but a lot of fun too!"
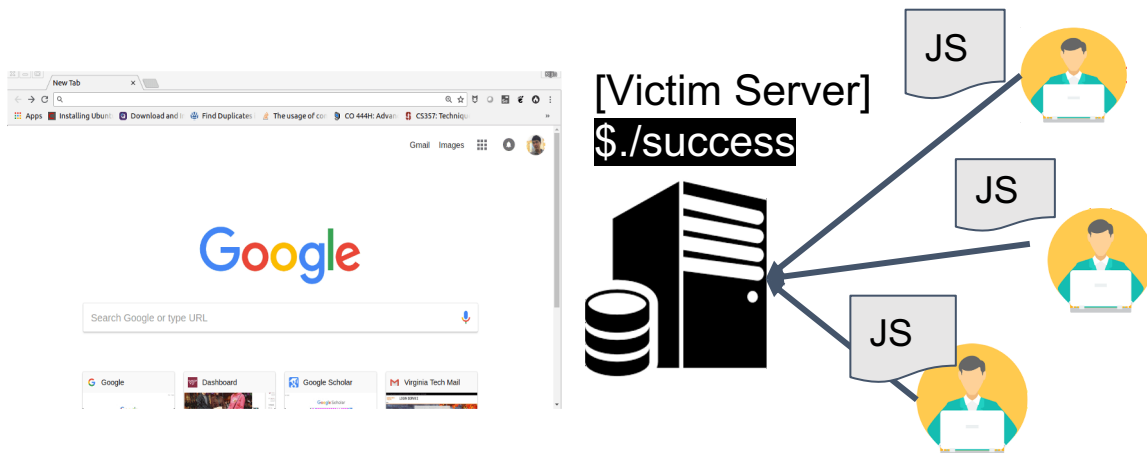
stewes36@vt.edu

Students' Achievements

Full Grades

Appendix: Understanding Heap Spraying Attacks

User Study (Pre/Post Survey): IRB, 540 undergrads for 3 Semesters
(see more detail in https://kjproj84.github.io/teaching)

- Next Level of "**Attack Lab**"
- Systems: Victim Server, Grader
  :Executing and Evaluating Programs
- Extension of JavaScript Engine: V8

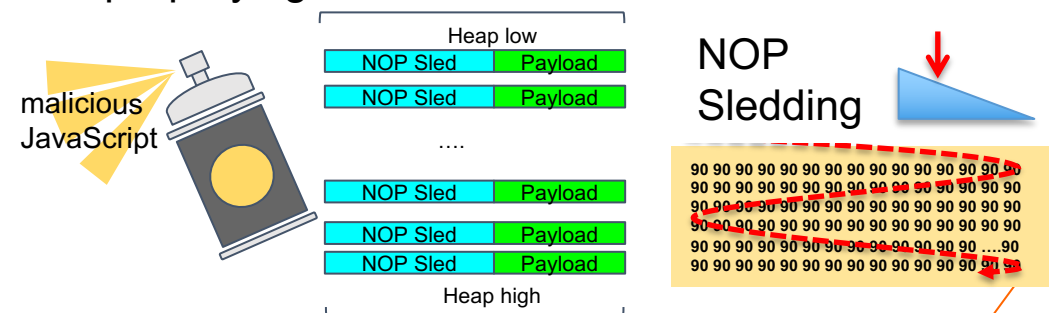**1. Crafting Assembly Code for a system call**

```
./sample: file format elf64-x86-64
Disassembly of section .text:
000000000400078 <start>:          "Payload"
400078:   48 c7 c0 01 00 00 00   mov $0x1,%rax
40007f:   48 c7 c7 01 00 00 00   mov $0x1,%rdi
400086:   48 8d 35 19 00 00 00   lea 0x19(%rip)…
40008d:   48 c7 c2 0e 00 00 00   mov $0xe,%rdx
400094:   0f 05                  syscall
400096:   48 c7 c0 3c 00 00 00   mov $0x3c,%rax
40009d:   48 c7 c7 00 00 00      mov $0x0,%rdi
4000a4:   0f 05                  syscall
```

[Victim Server]
$./success

3. Submitting JS Code to Attack Server

2. Heap Spraying with JS Code

malicious JavaScript

Heap low

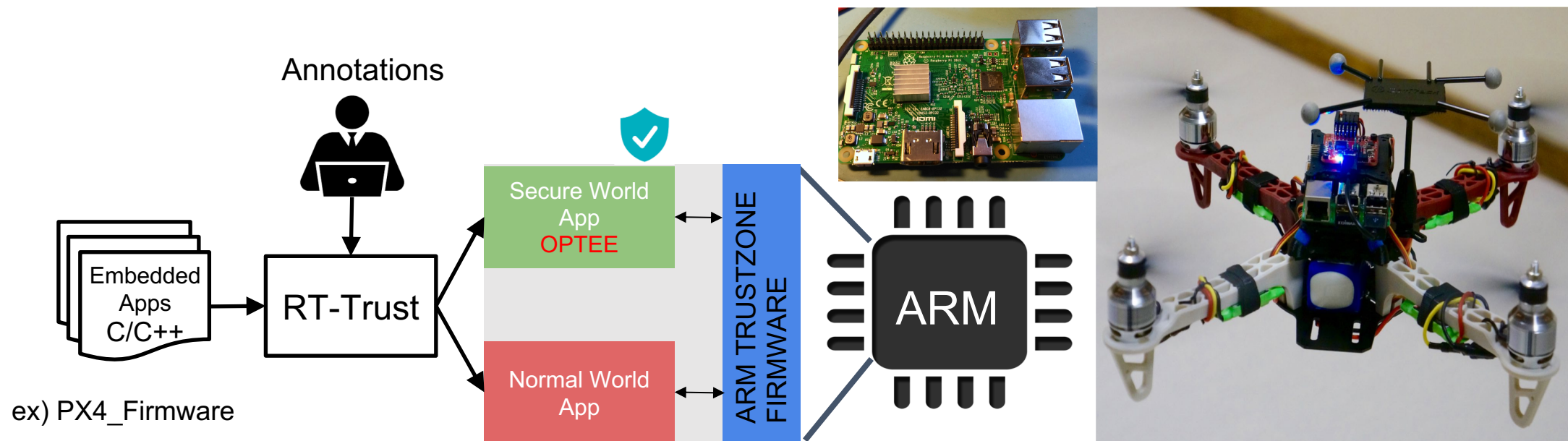| NOP Sled | Payload |
NOP Sled	Payload
....	
NOP Sled	Payload
NOP Sled	Payload
NOP Sled	Payload

Heap high

NOP Sledding

15

Appendix:
Refactoring Embedded Apps for Trusted Execution
[GPCE 2018], Best Paper Nomination at [COLA 2020]

- Programmers: **Annotating CPI** portions

- Partitioning C/C++ code into the <u>regular</u> and <u>trusted</u> parts

  - LLVM/Clang based Analysis/Refactoring, OPTEE (SGX in [COLA 2020])

ex) PX4_Firmware

Appendix:
Learning Translating Rules/APIs from Cross-platform Apps
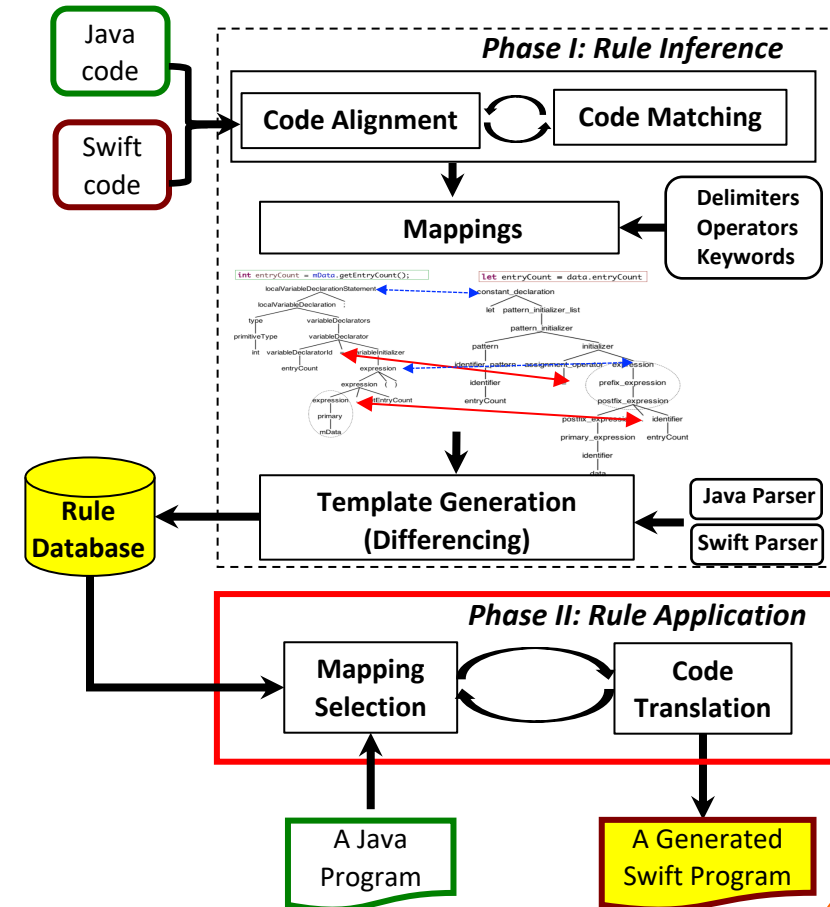### Best Paper Nomination at [MobileSoft 2018]

```java
//PieChart.java
1  public class PieChart extends PieChartBase<PieData>
{...
2    private void calcAngles() { ...
3      int entryCount = mData.getEntryCount();
4      int cnt = 0;
5    for(int i = 0; i < mData.getDataSetCount(); i++){
6    IPieDataSet set = mData.get(i);
7    ...}}}
```

```swift
//PieChartView.swift
public class PieChartView: PieChartViewBase
{…
    private func calcAngles() { ...
      let entryCount = data.entryCount
      var cnt = 0
      for i in 0 ..< data.dataSetCount {
        let set = data[i]
    ...}}}
```
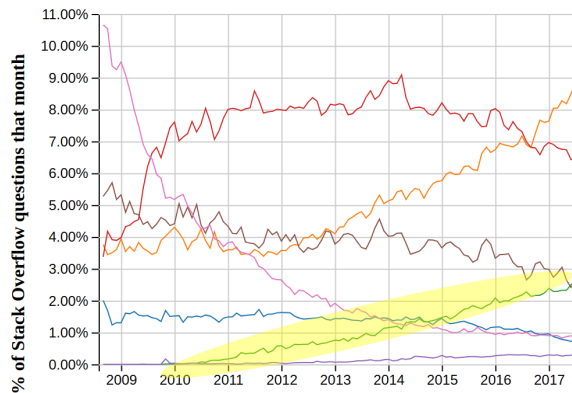
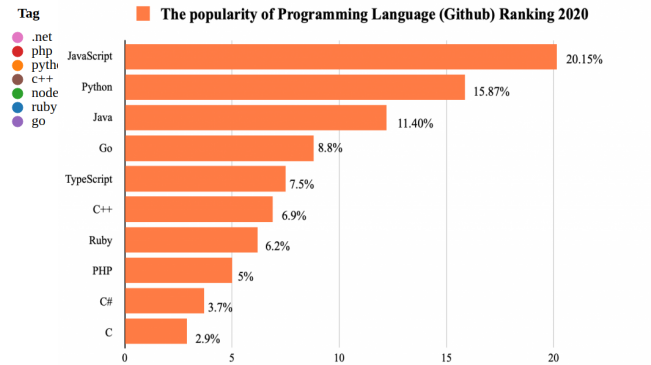| No.	Java Syntax type	Swift Syntax type	Java template	Swift template
1	typeDeclaration	class_decl	public class $p10 extends $p11 {...}	public class $p10: $p11{...}
2	classBodyDecl	function_decl	private void $p20() {...}	private func $p20() {...}
3	localVarDeclStmt	cnst_decl	$p30 $p31 = $p32;	let $p31 = $p32
4	expression	expression	$p33.getEntryCount()	$p33.entryCount
5	localVarDeclStmt	var_decl	$p40 $p41 = $p42;	var $p41 = $p42
6	statement	for_in_stat	for($p50 $p51 = $p52; $p51 < $p53; $p51++) {...}	for $p51 in $p52 ..< $p53 {...}
7	expression	expression	$p54.getDataSetCount()	$p54.dataSetCount
8	statement	cnst_decl	$p60 $p61 = $p62;	let $p61 = $p62
9	expression	expression	$p63.get($p64)	$p63[$p64]

17

VIRGINIA TECH

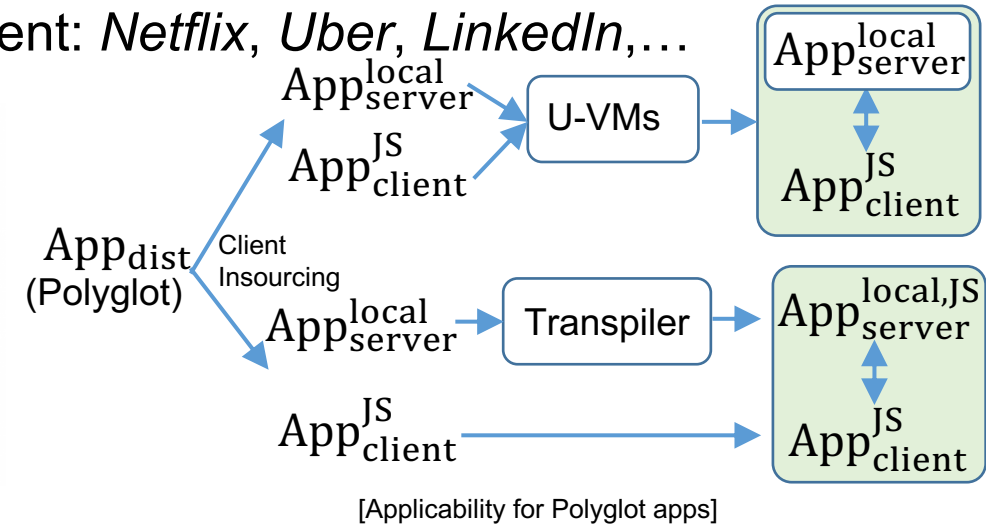# Appendix: Applicability of Client Insourcing

- **Subject**: Full-Stack JavaScript apps (Node.js): Popular in Backend and Open Source

    - Backends follow a lower load time development: *Netflix, Uber, LinkedIn,...*

[Trend in JS Backend Developments]

[Trend in Github Repositories]

[Applicability for Polyglot apps]

- **RESTful HTTP Protocols**

- **Insourcing Business (Application) Logic only**

    - What else? *Failure/Exception handling Logics*

- **Server State Isolations/Replications**

    - Database with **SQL**, **Files**, and **global variables**

(Server-side)

(Client-side)

Business Logic

Programmer-written Code

Middleware Libraries

Programmer-written Middleware API Calls

18