Unhackable Across 30 Years, End in Sight

Blaine Garst, Founder, **Wizard**

The Planet Earth Society Inc.
A CA Social Purpose Corporation

Feb 5 Agenda: Unhackable

- **Now**: solve identity on Raspberry Pis (**Save** Planet: TheDew)

Act V

- **Soon**: Unhackable **Hardware** & **Software** Overview (TheDew)
    - SQueue: lockless double ended queue SM.ART dispatch hypervisor-ish
- **Past Saves**: SVR4 UNIX® => => => POSIX, NeXT, Apple, WG14 C

Act I

- **SVR**2.1,3,**4** Unix®: => Solaris, **Spring**, **fizix**, **POSIX**, Object Management Group (COBRA)

Act II

- Mach Kernel & NSObject/Language Runtime Architect: NeXT-Apple (**PDO**, **ECC**)

Act III

    - Objective-C “Interfaces”, Frameworks => (**Java, C#, Android)**, ^Blocks, ARC, (llvm, clang)

Act IV

- WG14 (C: C11 _Atomic (**weak memory model**), C17; FPE, CSCR) defect fix “wrangler”

Unhackable: Big Picture

Thesis

- Measurable as fitness-to-purpose on distributed RT platform, needs
  - Unhackable {**Identity**+**Hardware**+**Software**+Data+Networking}
- Today: Hackable {Identity+Hardware+Software+Networking}
  - Solution: rewrite that which I… instigated.
  - Requires Actor Language Algorithms embodiment (as Unix required C)

Unhackable: TheDew Identity

Trusted identity can only come from your active friends and family

- Raspberry Pi3,4 **DewDrop**s plugged into your router, UDP 12345 only
- Form private all-signed tunneled network among your **bestest** friends
- New “whitepages” (CA) protocols for key rolling, validation (future RFCs)
- Share non-commercial local free music, code…, no money, no sniping
  - Do collaborative party planning & have FUN! (Social Purpose Network)
- Content via signed self-contained “tarballs”, lazy updates with your inputs

Unhackable: Hardware (1/2)

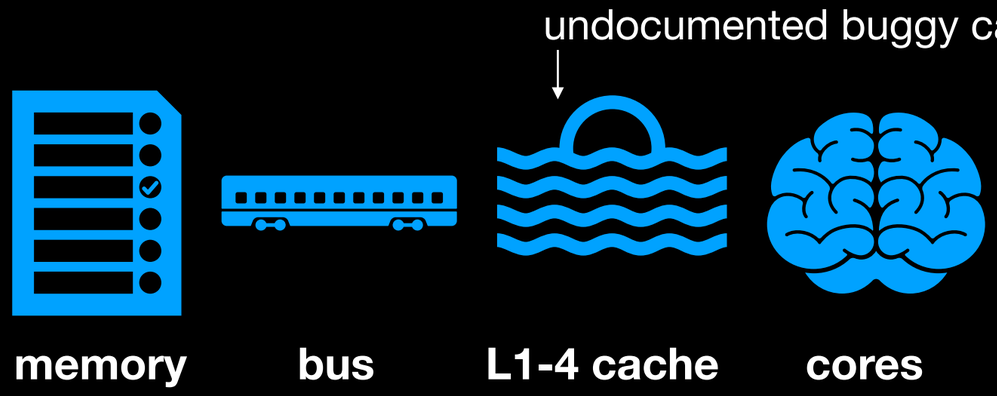
Prereq: **Guarding Against Physical Attacks: The XBox One Story, *Tony Chen, Microsoft***
https://www.platformsecuritysummit.com/2019/speaker/chen

- Axel Kloth, Abacus-semi.com, supercomputer chip-set w Smart Memory

Solves:
- 2016: “most secure” ever seen, USSOC eval (reboots to known state, updates, 99.999%)
  https://en.wikipedia.org/wiki/United_States_Special_Operations_Command

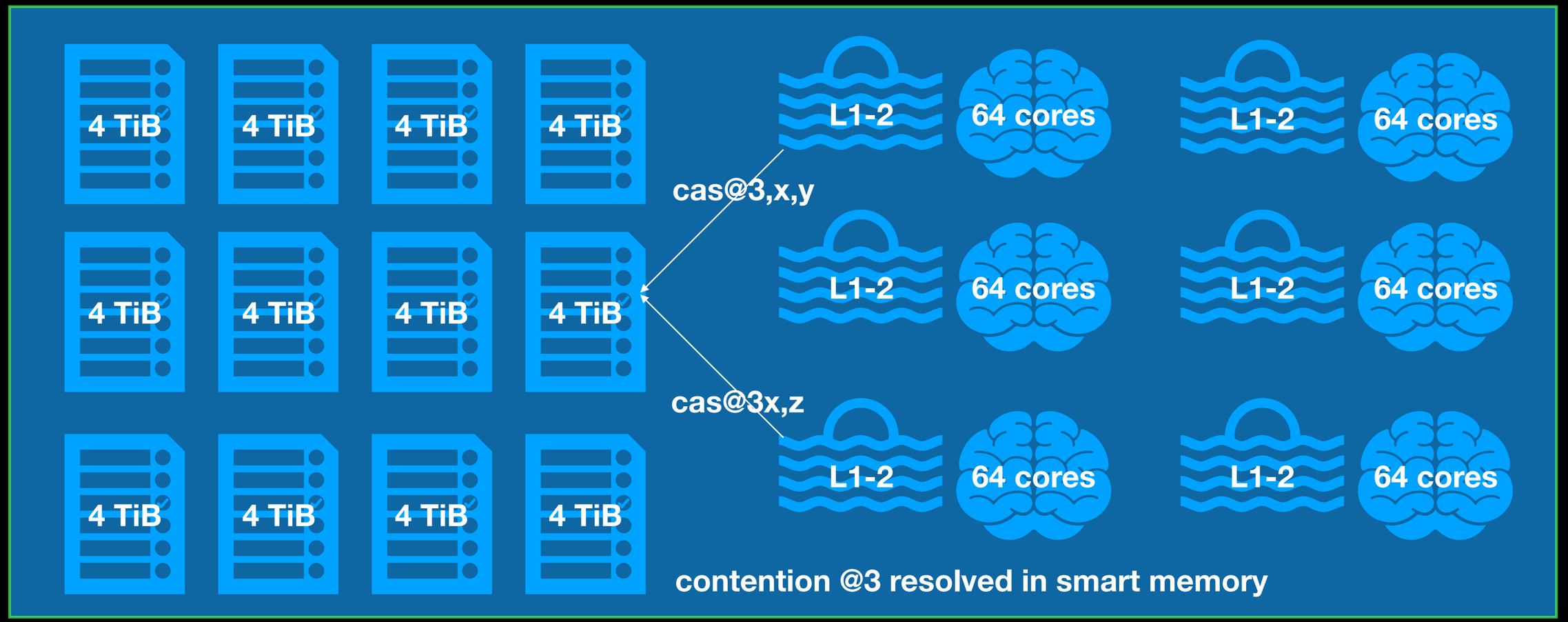
Solves:
- Von Nuemann bottleneck at bus

Solves:
- contention/“weak ordering” heat chaos in caches.



Unhackable: Hardware (2/2)

- Axel Kloth, Abacus-semi.com, supercomputer chip-set w Smart Memory
- resolve contention in Smart Memory on pico-switch multi-hop fabric, not bus



Unhackable: Software (1/3)

GOAL

- Turn Alchemeic “function/modules” into Periodic Table of Actor Algorithms

imperative: {C, Java, Python, ..} functional: {Haskell, ML} logic: {Planner, PROLOG}

all boil down to Actor HLIR ASTs

gcc	**clang**	**icc**	**clangX**
lex1	lex1’	lex2	lex3
parse1	parse2	parse3	parse3’
gen1	gen2	gen3	gen4
assm1	assm2	assm2’	assm3

what if,
all lexers,
all parsers,
all codegens,
all assemblers,

inter-operated cleanly?!

Unhackable: Functions (2/3)

- C Functions fraught with side-effects, UB, global dependencies, … JUST SAY NO!
- Actor Functions are Lambdas with Actor RunTime accumulated work
  - FUNCTION (a b) [SEND $println OPER a + b OPER; SEND;] FUNCTION;
  - The $println actor is a USES implicit parameter,
    - FUNCTION (a b) [uses: $println] SEND … SEND; FUNCTION;
      - *{ type(type(a)+type(b)) } => $println* more technically
    - fully parameterized, forms parametric signature “key” to all implementations

Unhackable: Software (3/3)

GOAL

- Turn function/modules “Alchemy” into “Periodic Table” of Actor Algorithms

gcc	**clang**	**icc**	**clangX**
lex1	lex1’	lex2	lex3
parse1	parse2	parse3	parse3’
gen1	gen2	gen3	gen4
assm1	assm2	assm2’	assm3

lex(input: *charstream*, tokens: *pattern*)
=> *tokenstream(*pattern*)

parse(input: *tokenstream(*pattern*)*,
***grammar(tokens: *pattern*, NT, start)*)**
=> *AST(grammar….)*

generator(ast: *AST(grammar…)*, arch: *arch*)
=> *LLIR(arch: *arch*, 32)*

assembler(llir: *LLIR(arch: *arch*, *{32|64})*)
=> binary(*arch*, *{32|64}*)

Unhackable: Software (4/4)

PROPERTIES OF PERIODIC TABLE OF ALGORITHMS

- Algorithms not patentable yet can be Trade Secret IP
- random walk down produces correct result
- any module can be replicated & run against itself & others & joined by an actor that implements any comparison policy it wants (perhaps on other cpus)
- least energy algorithms move module “leftward”, forming a self-optimizing for least energy platform (energy = cpu + memory + network)
- every module signed with ownership and non-commercial use contracts,

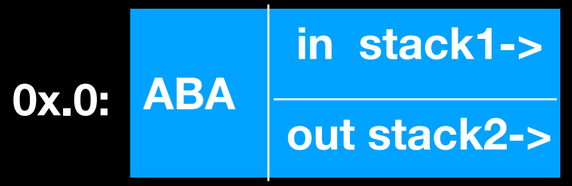
goal: commercial use contracts optional, extra gov. access layer

  - allows refugee camp coders an economic platform, etc.,

2012 SQueue lockless data structure and algorithm

SQueue: lockless queue (1/2)

- CMPXQ (16 byte compare & swap) available: IBM, sparc, x86, arm…, not RISC-V
- ABA: pack ptr to squeeze out alignment, combine with unused high bits
    - increment (~24bit) ABA counter on every CMPXQ high level operation



ptr1: ABA pushdown stack of incoming work using hole in work item

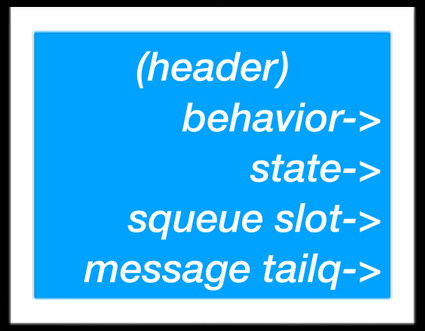
ptr2: ABA popup stack of outgoing work

- dequeue: while ptr2 == NULL, ABA_cmpxq ptr1,ptr2; ABA_cmpxq pop
- order skewed but guaranteed progress, good enough for core scheduling

SQueue: SM.ART (2/2)

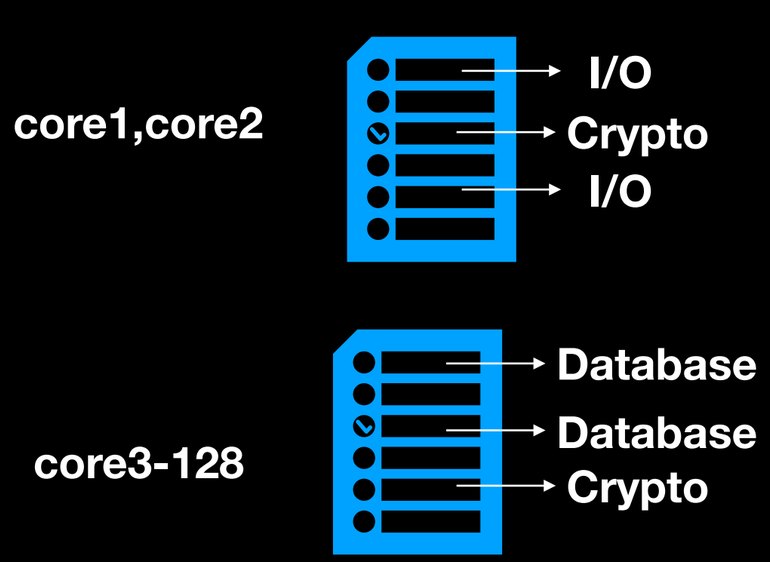
- **S**ymmetric**M**ulti{core/thread/processor?} . **A**ctor **R**un-**T**ime
- parallel: { cores } while (a=abadq(runL[core])) while (msg=dq(a.tailq)) msg=>a

SM.ART actor layout



Actors only ever on 1 run-SQueue

{cores} while(a = dq(runL[core])
while(msg=dq(a.tailq)) msg=>a



Run List of SQueues

(fed by ISR!)

System Wide SQueues

PAST (one liners)

Act I	• 1987 Saved UNIX, by accident!
Act II	• 1990-1996 Saved NeXT, sold sources that became Java
Act III	• 1996-2011 Built Apple (example: 10.6 Snowleopard on ^Blocks)
Act IV	• 2011-17 Saved C (no syntax for atomics was untenable), C17 bug fixes
Act V	• 2012-22 Saving the Planet with TheDew. Better together! Join us!

PAST: one liner highlights

Act I

- **SVR**2.1,3,**4** Unix®: => Solaris, **Spring**, **fizix**, **POSIX**, Object Management Group
- NeXT,Apple Mach Kernel & NSObject/Language Runtime Architect (MacOS, iOS, watchOS)

Act II

- DO, **PDO**, **ECC, Frameworks, Foundation.Framework NSObject**, retain/release/autorelease, -finalize; NSRunLoop, @try @finally, … ^Blocks, ARC, (llvm, clang)

Act III

- Objective-C sources sold to Sun, (stripped dynamism) => **Java => .Net, Android**
- **OO Multiple Inheritance of Interface, (not C++ implementation), broke IBM/industry 50 year business model of custom software on custom hardware**

Act IV

- WG14 C: C11 _Atomic (**weak memory model**)
- WG14 defect fix “wrangler” for C17, IEEE Floating Point, C Secure Coding Rules

[ACT I: How I Saved UNIX from UNIX® without even trying!]

- 1974 **University High School Graduate**, Urbana IL (then **PLATO**, now TheDew, they're in!)
- 1977-08-15 Bell Labs MTS, **Bell Data Network**, (MSCS Programming Languges and Systems, UCLA)
- 1984-09-10 transfer: Mgr Unix **System V 2.1** Networking: STREAMS, OSI TLI layer 4

  (1/2 Unix® market) System V 2.0: 14 character files, classic triple inode fs, single cpu
  (1/2 market no-fee) dozens of Unix competitors (SunOS, DEC ultrix, …)
  Tandem, DEC VMS, Apollo, Pyramid, Data General, Novel, …, MICROSOFT, IBM, …

- 1987-(spring!) acquired & integrating VFS, NFS, symlinks from SunOS, /proc from MH research (SVR3.1??)
  - SVR4… under **NDA** 2 DAYS with BILL JOY, merged in SunOS/BSD syscalls!
- 1987-10-19 **ATT buys 20% Sun**, hw: **sparc**, sw: **merged** unix (**Phase I**), **SVR4**, (license fee UNIX® for all), SunOS=>Solaris

  www.nytimes.com/1987/10/19/business/at-t-deal-with-sun-seen.html

Sun Servers + ATT workstations!
Everybody pays UNIX® fees!

INDUSTRY HORKED!

- 1988 **Hamilton Group** formed => Object Management Group (UML, CORBA)
- 1988-1989, (**Phase III**) Bell Labs, Menlo Park, Bell Labs 3 + SUN Labs 3, **"Spring"**
- 1989-1990-07-31 Bell Labs, Menlo Park, **"fizix" register xfer channels (doors) nanokernel**
- ATT ditches, donates SVR4 spec to /usr/group standards track => POSIX => **license FREE clean room UNIX**

ATT BAILS!

FIZIX - 1989, WE 3B2

My one and only C++ program (“C++ is bad cheese!” - Bertrand Serlet)

- Address Space Domains, not processes
- Wandering threads (a la V System, David Cheriton)
    - threads carry cpu limits (Actors)
- IPC in registers capability channels is only primitive
    - transfer capabilities in messages like Mach (Actors)
- Performance: IPC @10x C++ vtable dispatch (in prototype) - fast enough for multi-kernel!

MACH IPC way too slow, Avie Tevanian@NeXT agreed, I got hired!

    - Upcalls to domain context unspecified (Actor messages!)

The End

```
{
  [ AFTER $println (applause) 2000  AFTER; ]
  [ SEND $println That’s all Folks! SEND; ]
}
```

I will code & teach Hactorscript/TheDew on TheDew to limited (~24) participants,
email “FOSDEM IN” to blaine@theplanetearthsociety.com before 2022/02/06 17:00 (24hr window)
after 2022/02/05 17:00

PDF: Two more things…

profound influences

- 1645 A Book of Five Rings, Myamoto Musashi
- OHMU, partial evaluation, DeLesley Hutchins, he got mutability wrong, as I did @ NeXT
    - 2003 OOPSLA03 The Power of Symmetry: Unifying Inheritance and Generative Programming, (paywall, worth it)
    - 2009 University of Edinburgh, Ph.D. thesis, Pure Subtype Systems: A Type Theory for Extensible Software
- 1974 **Carl Hewitt**, Viewing Control Structures as Patterns of Passing Messages, MIT A.I. Memo 410
- 2012 **Dale Schumacher**, Actor Idioms, AGERE! 2012, Tucson, AZ, and Humus, various
- 2021 Makerspace @ Uni (Urbana, IL) https://www.youtube.com/watch?v=VuRj2PSK0Ek
- 2022 FOSDEM obligatory fix-point self-reference https://fosdem.org/2022/schedule/event/bgarst/

WG14 C, IEEE FPE, CSCR selected papers

http://www.open-std.org/jtc1/sc22/wg14/www/wg14_document_log.htm

N1888 2014/11/05 Garst, Discussion of C11 DR452
N1864 2014/09/22 Garst, Proposed Resolution of DR431 atomic compare-exchange of struct
N1863 2014/09/22 Garst, Proposed Resolution of DR423 Underspecification for Qualified rvalues
N1832 2014/04/25 Garst, TS 17961: *C Secure Coding Rules* Defect Reports
N1824 2014/04/08 Garst, initializing using `ATOMIC_VAR_INIT`
N1804 2014/03/15 Garst, Discussions on DR440, DR441, DR442, DR444, and DR445
N1803 2014/03/15 Garst, Atomic issues in DR423 and DR431
N1736 2013/08/02 Garst, Use byte instead of character for memcmp(), memcpy()
N1623 2012/06/08 Garst, Late comments - N1609
N1567 2011/03/17 Garst, footnote 113
N1536 2010/11/04 Garst, _Atomic Qualifier Issues
N1530 2010/11/03 Garst, Atomic Bitfields Implementation Defined
N1526 2010/10/14 Garst, Atomic C1x/C++0x Compatibility Refinements
N1525 2010/10/12 McKenney, (Garst) Memory-Order Rationale
N1522 2010/10/11 Garst, Atomic Refinements
N1485 2010/05/26 Garst, Atomic Proposal - Draft 10
N1476 2010/05/24 Garst, Atomic Proposal (v6)
N1473 2010/05/10 Garst, Atomic Proposal (version 5)
N1457 2010/04/27 Garst, Blocks presentation
N1456 2010/04/27 Garst, What's New in Objective-C presentation
N1452 2010/04/12 Garst, Atomic Proposal
N1451 2010/04/12 Garst, Blocks Proposal
N1370 2009/03/10 Garst, Apple's extensions to C

USPTO Issued Patents

Extended Garbage Collection
Gerald Blaine Garst, Jr
8504596, Aug 6, 2013, Apple, Inc.

Method & apparatus for enforcing software licenses
Gerald Blaine Garst Jr, Bertrand Serlet
6188995 February 13, 2001, Apple Inc.
8027925 September 27, 2011, Apple Inc.
8452712 May 28, 2011, Apple Inc.
8781971 July 15, 2014, Apple Inc.

Memory Management for Closures
Gerald Blaine Garst, Jr., William Bumgarner, Fariborz Jahanian, Christopher Arthur Lattner
8341614 December 25, 2012, Apple Inc.

Per-thread garbage collection,
Gerald Blaine Garst, Jr., Gregory Robert Parker, Douglas Joshua Behnke, Patrick C. Beard
7991808 August 2, 2011, Apple Inc.
8255436 August 28, 2012, Apple Inc.

Fast function call dispatching
Steve Naroff, Blaine Garst, Greg Parker
8291395 October 16, 2012, Apple Inc.

Transparent local and distributed memory management system
Blaine Garst, Ali Ozer, Bertrand Serlet, Trey Matteson
5687370 November 11, 1997, NeXT Software Inc;
6026415 February 15, 2000, NeXT Software Inc.
6304884 October 16, 2001, Apple Inc.
6571262 May 27, 2003, Apple Inc.
7305538 December 4, 2007, Apple Inc.
7716450 ay 11, 2010, Apple Inc.

Method and apparatus for fast elliptic encryption with direct embedding,
Richard Crandall, Blaine Garst
6307935 October 23, 2001, Apple Inc.