



Windows Vista Heap Management Enhancements

Adrian Marinescu
Development Lead
adrmarin@microsoft.com



Agenda

- Windows NT Heap Management basics and evolution
- Windows Vista heap – major milestone
 - Development principles and guidelines
 - Security features
 - Performance features
- Q & A



Introduction

- Security – industry-wide concern
- TwC driving multiple security initiatives
- The NT Heap
 - Strategic point in defense
 - Improved to respond to industry trends in usage

Part I – Basics



Heap Evolution

Basics	Performance	Opt-in SMP Scalability Heap Mitigations	Enhanced security Performance Quality tool	Time →
NT 4	NT 4 / SP4 Windows 2000	XP / SP2 Windows 2003	Windows Vista	

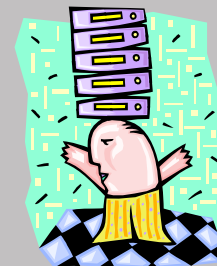
Industry



Workload



Exploitation

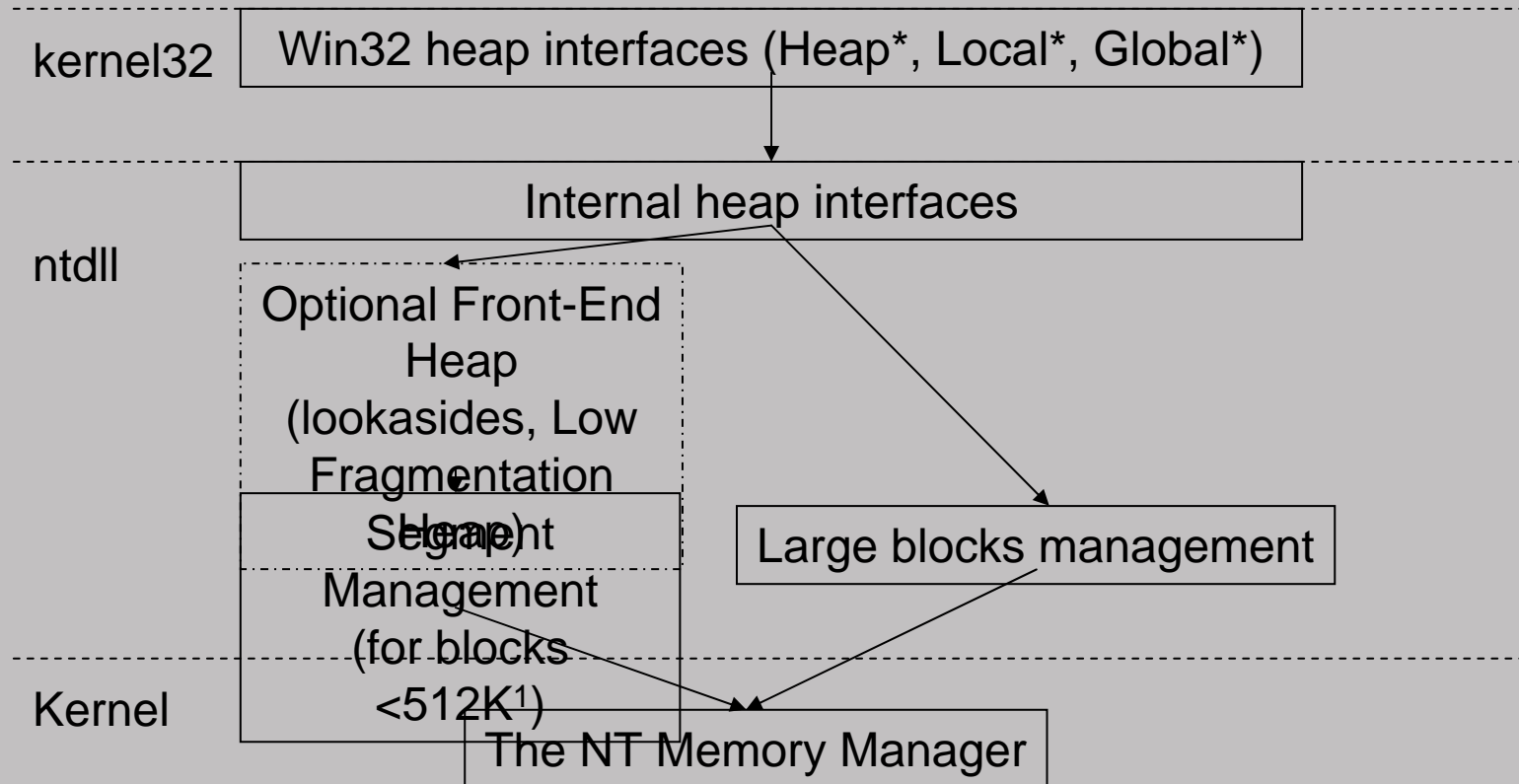


Parallelism



NT Heap Overview

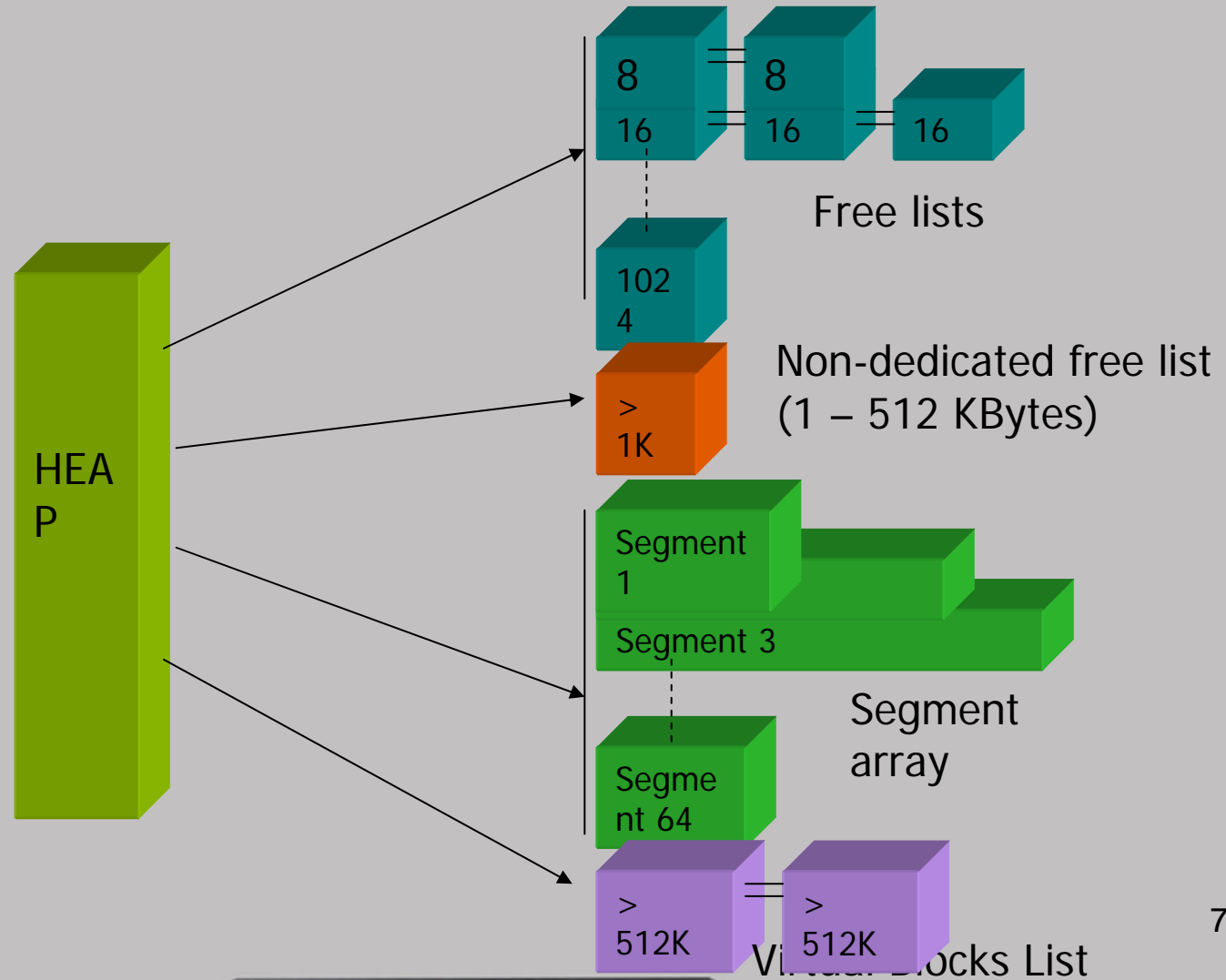
Windows applications



¹ On 32 bit platforms

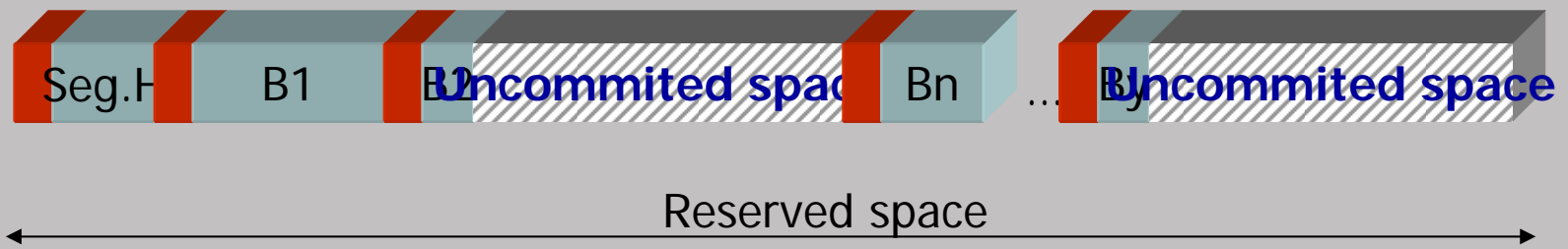


The NT Heap Core Management





Heap Segments





Block Entry in prior Windows NT Versions





Role of Link Entry in Early Exploits

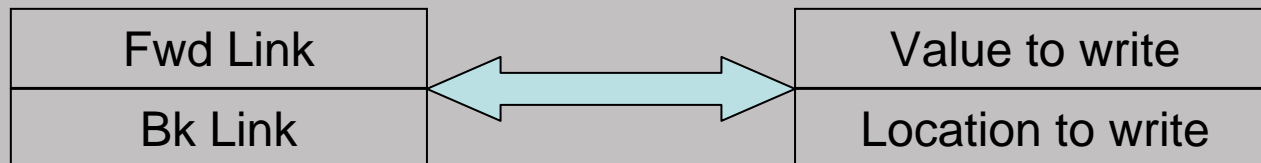
- Arbitrary pointer write

```
mov eax, DWORD PTR [ecx]
```

```
mov ecx, DWORD PTR [ecx+4]
```

```
mov DWORD PTR [ecx], eax
```

```
mov DWORD PTR [eax+4], ecx
```

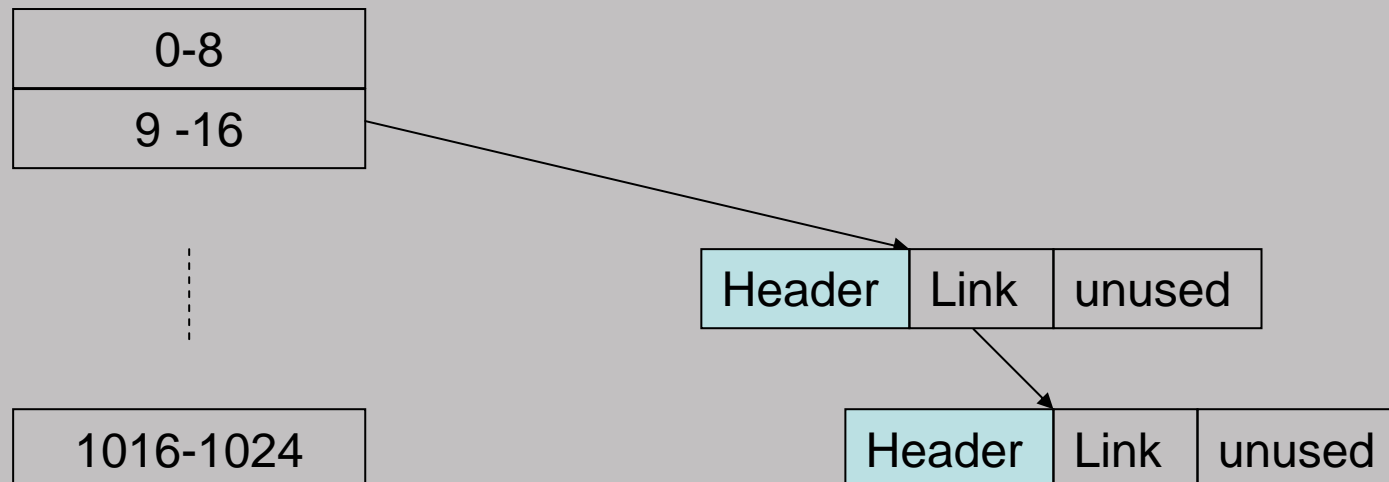




Lookaside Lists

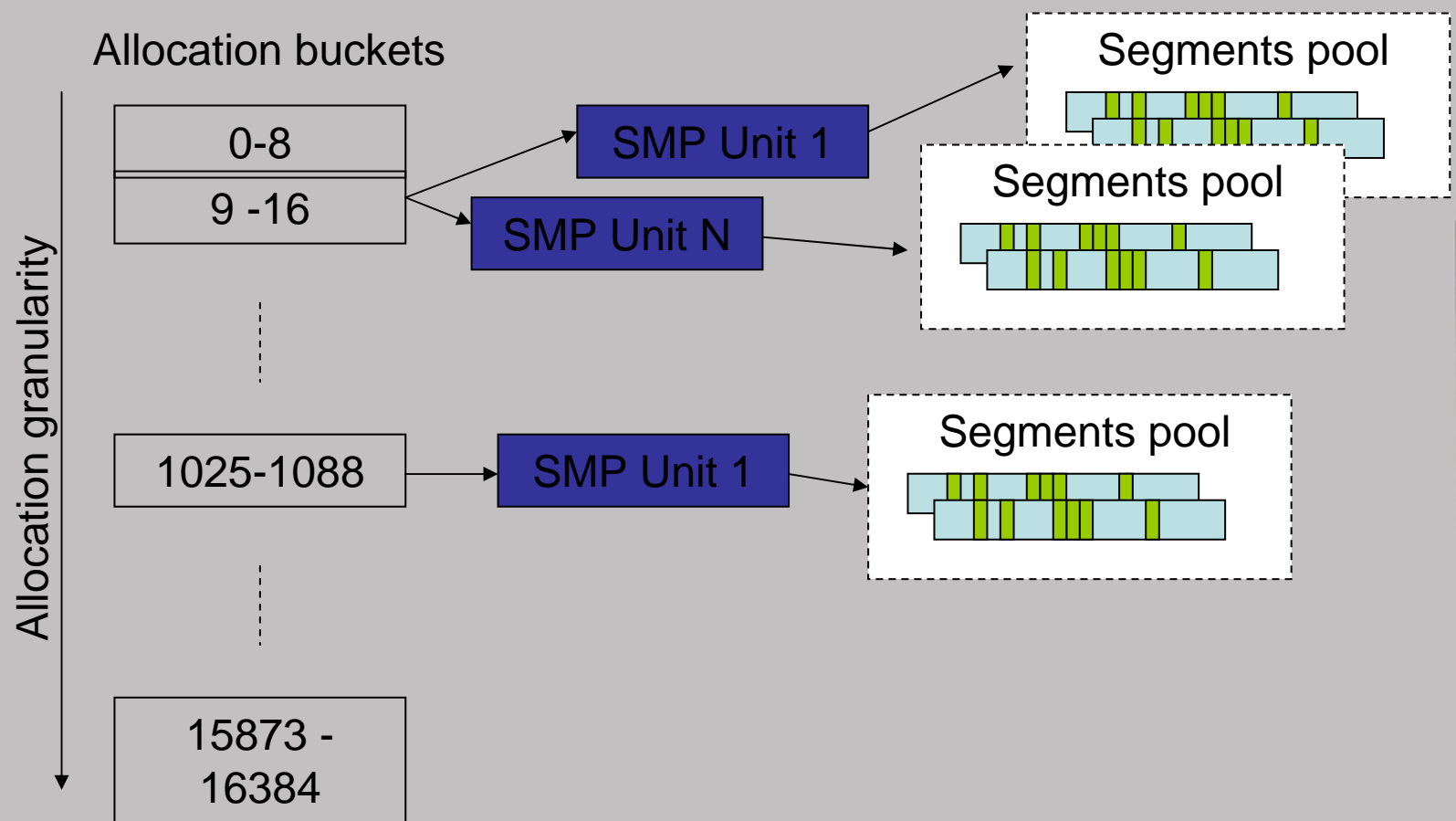
- Non-blocking single-linked lists

Lookaside lists



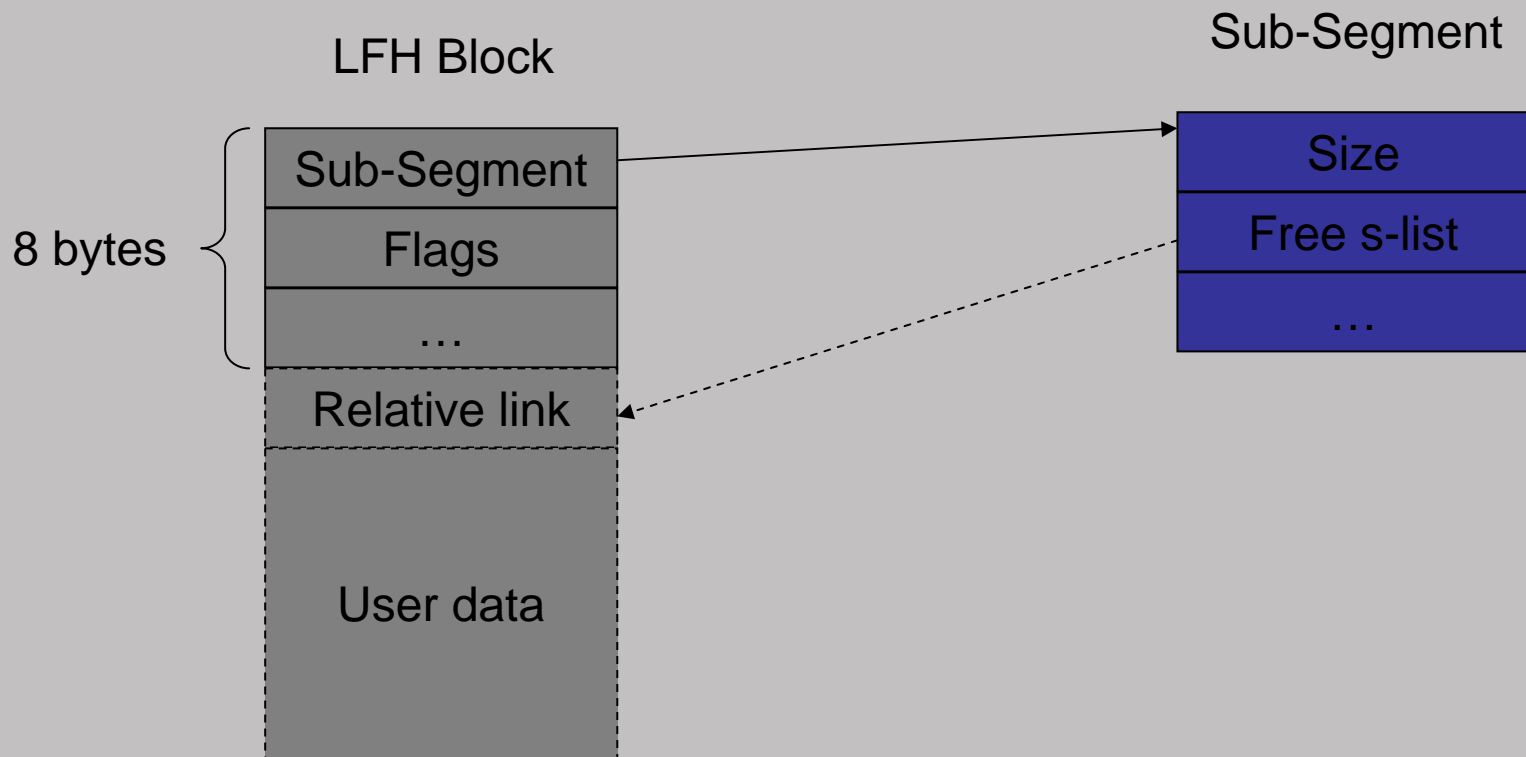


The Low Fragmentation Heap Architecture





Low Fragmentation Heap Block Entry





Early Heap Mitigations

- Safe List Removal

Entry->FwdLink->BkLink == Entry->BkLink->FwdLink == Entry

- 8-bit cookie tested on free
- LFH block entry encoding

F (random number, Block address, heap)



Change in Landscape

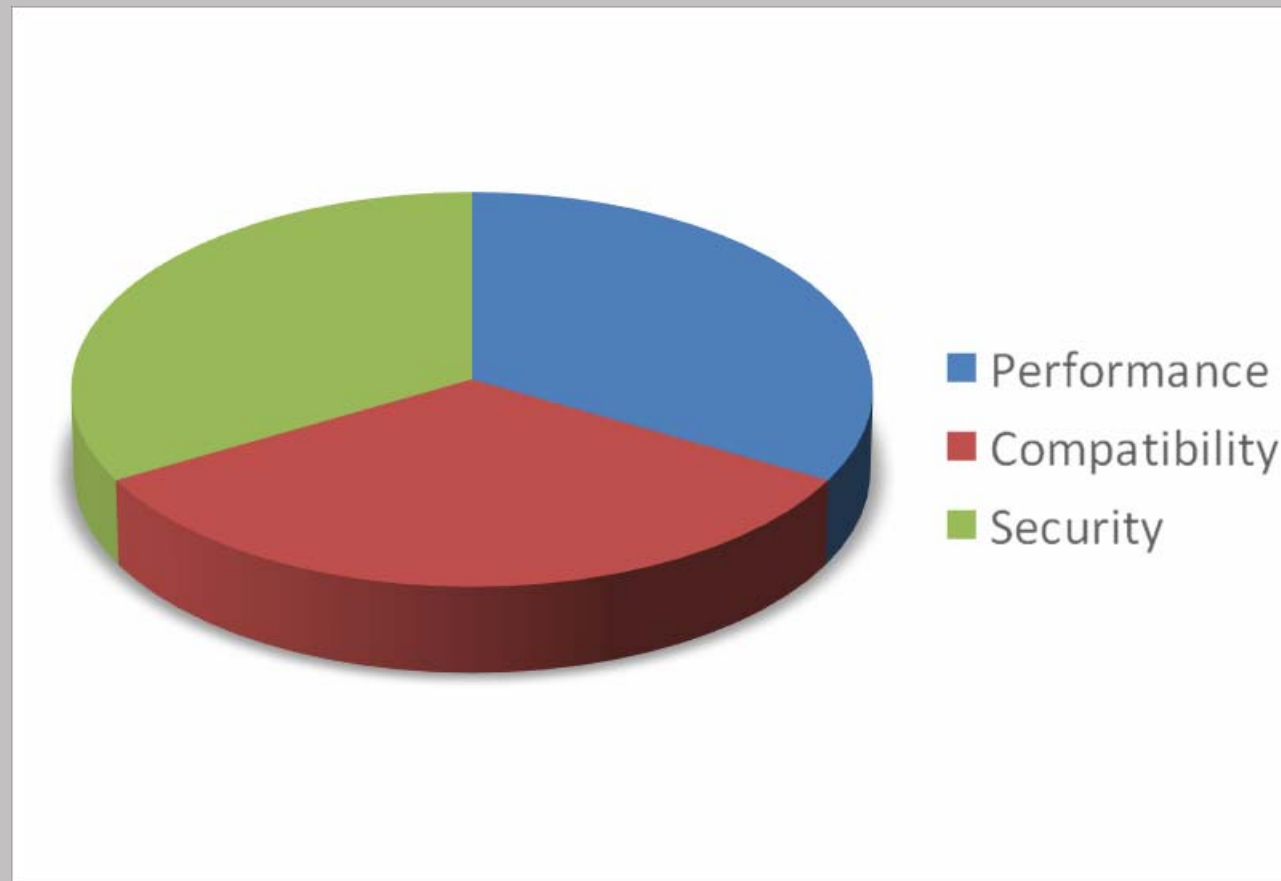
- New exploiting methods surfaced
- Change in usage outlook
 - Memory usage
 - Increase availability of SMP
 - Increase relevance of 64 bit computing
- Code quality – higher demand in industry



Windows Vista Heap Manager Key Development Directions

- Performance and reliability
- Security
- Code quality

Windows NT Heap Requirements





Security

- Correctness – like:
 - Guarantees requested sizes
 - Lifetime of allocations
 - Clearing content when requested etc.
- **Defense line in heap based exploits:**
 - Attempts to mitigate the effect of an attack
 - Makes difficult hiding heap-based exploits



Performance

- Scale from small devices to large servers
- Optimized for varied usage patterns
- Follow the industry trend
 - Memory usage
 - Increase in SMP availability
 - H/W architecture advances



Compatibility

- Applications may rely on things like:
 - Realloc returning same pointer
 - Read/write after releasing a block
 - Double free
 - Overruns over unused structures etc.
- Heap changes may have unintended effects, such as:
 - Crashes, leaks or broken functionality in poorly written applications
 - Severe performance regressions

X'col 2006



Part II - Windows Vista Heap



Windows Vista Heap Security Features

- Block metadata randomization
- Integrity check on block entry
- Algorithm variation in response to usage pattern
- Random rebasing
- Function pointer randomization
- Abrupt application termination on error



Block Metadata Randomization

- A part of the header is XORd with a random value
- Low performance impact
- Should make guessing the right value impractical
- Flexible and contained algorithm and implementation
- Agile in updates



Entry Integrity Check

- Previous 8-bit cookie has been repurposed to validate a larger part of the header
- Value may be randomized along with the other fields
- Validated during internal operations too

X'col 2006



Demo – Heap Header Layout



Runtime Algorithm Variation

- Automatic tuning
 - Shift to LFH allocations at arbitrary points on runtime
 - Triggers on various patterns
 - Involves also de-commit / commit policies



More Heap Randomizations

- Heap base randomization – things to consider:
 - Fragmentation of the application address space affecting large server applications
 - Possible performance issues if higher randomization is used
- Heap function pointer randomization
 - Takes away a known place to facilitate the code execution along with rebasing

X'col 2006



Demo



Abrupt Termination on Error

- Any data inconsistency or invalid heap function usage detected may trigger it
- The scope is process-wide (any heap in the process has the same behavior)
- The process is terminated via Windows Error Reporting
- Detailed info is available in the dump file
- No function provided to disable it
- On by default for 64 bit platforms & apps



Termination on Errors (cont.)

- Programmatic opt-In method
(new **HeapEnableTerminationOnCorruption** class defined)

```
BOOL HeapSetInformation(  
    HANDLE HeapHandle,  
    HEAP_INFORMATION_CLASS HeapInformationClass,  
    PVOID HeapInformation,  
    SIZE_T HeapInformationLength);
```

- Large number of components with Windows Vista are opted in
- The information is available in a debugger extension

X'COLI 2006



Demo



NT Heap Manager – Improves Code Quality

Benefits to app developers

- Early error detection
- Improved debugging aid to reduce cost of investigating corruptions
- Reduced tolerance to misuse
- Windows Vista apps will be more resilient to future heap changes



Known Attack Vectors & Windows Vista

- Removed lookaside list and array of lists targeted by previous exploits
- Integrity check on block metadata significant obstacle to brute force attacks
- Most Windows processes terminate on memory errors
- Dynamic (runtime) change in heap algorithms obstacle to consistent exploits
- Heap structures and memory mgmt changes limit portability of exploits



Security enhancements are a journey

- Mitigations are not substitute for good development practices
- Windows Vista is just a milestone in continual heap improvements



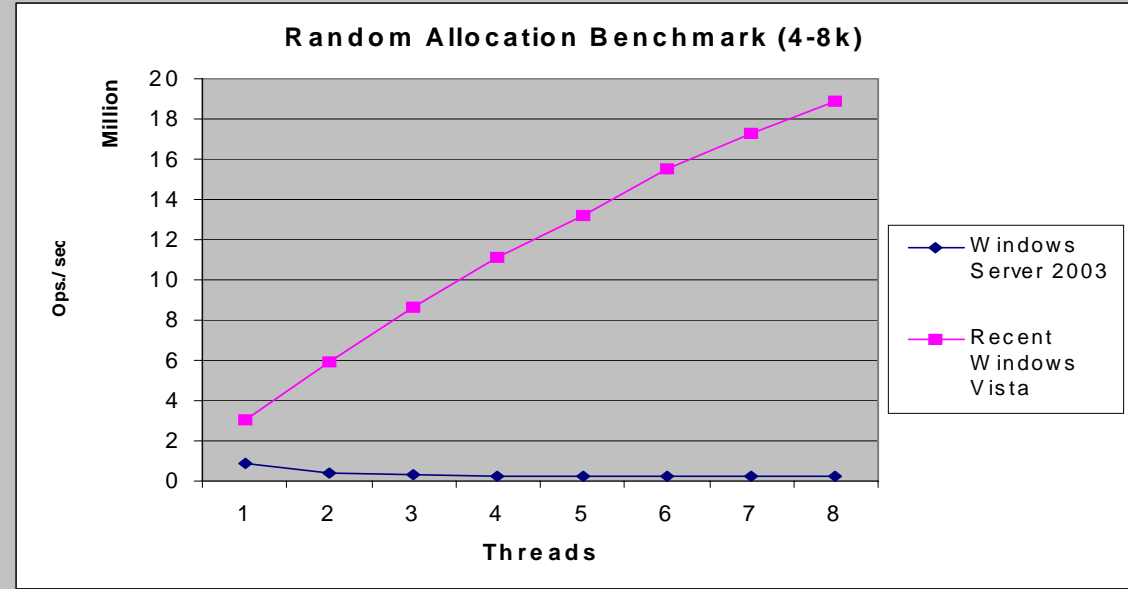
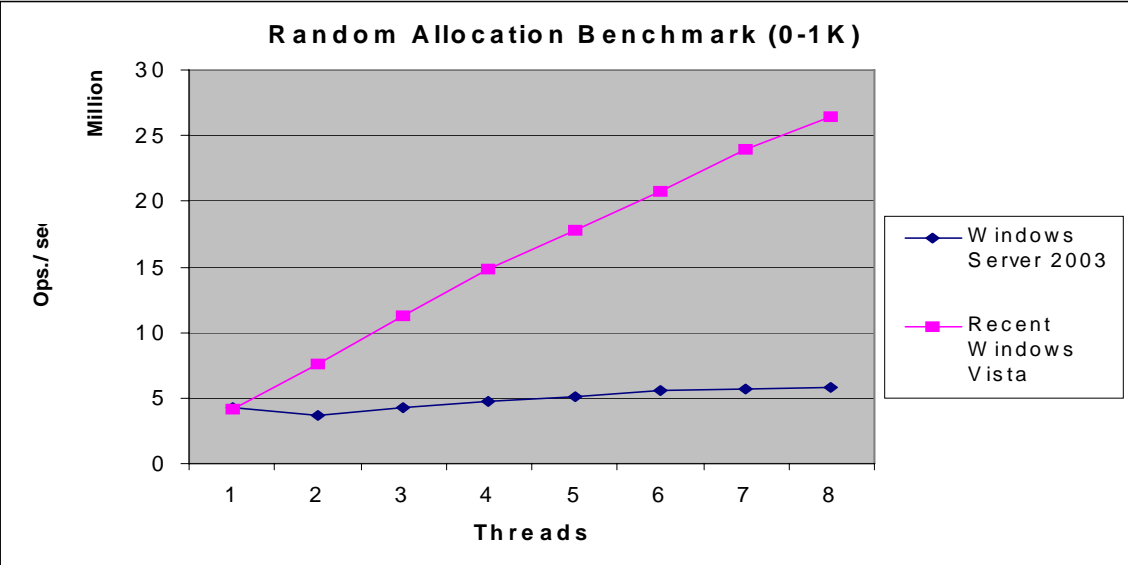
Windows Vista Heap Perf & Reliability

- Improved scenarios by default for:
 - SMP scalability
 - External fragmentation
 - Large heaps
- Improved reference locality on 64 bit platforms
- Reduced Virtual Address exhaustion
- Increased resilience to patterns involving long-term allocations



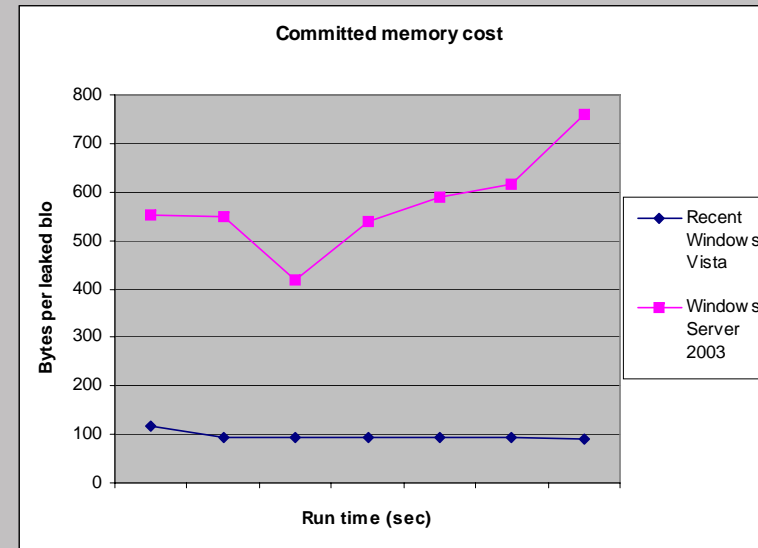
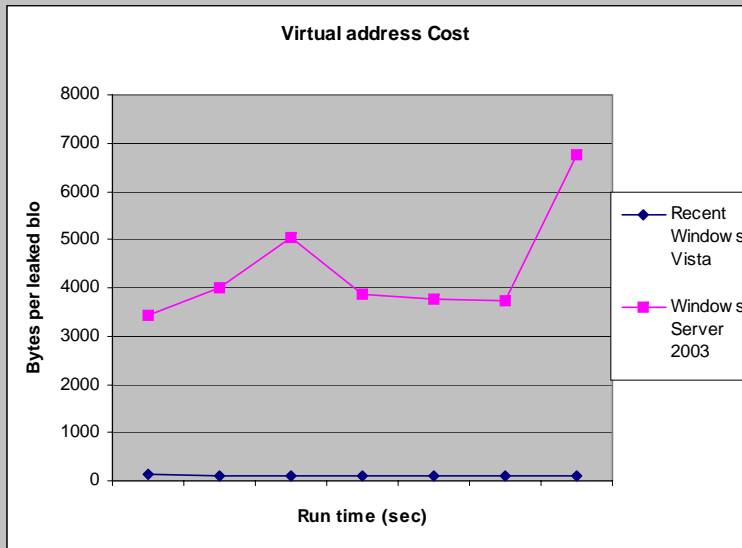
Key Performance Enhancements

- Automatic tuning
- Lower granularity of control policies to switch to the Low Fragmentation Heap
- Use of lazy initialization
- Redesigned segment management
- Improved internal lookup algorithms
- Addressed fragmentation in problematic scenarios
- Lower overhead on 64 bit



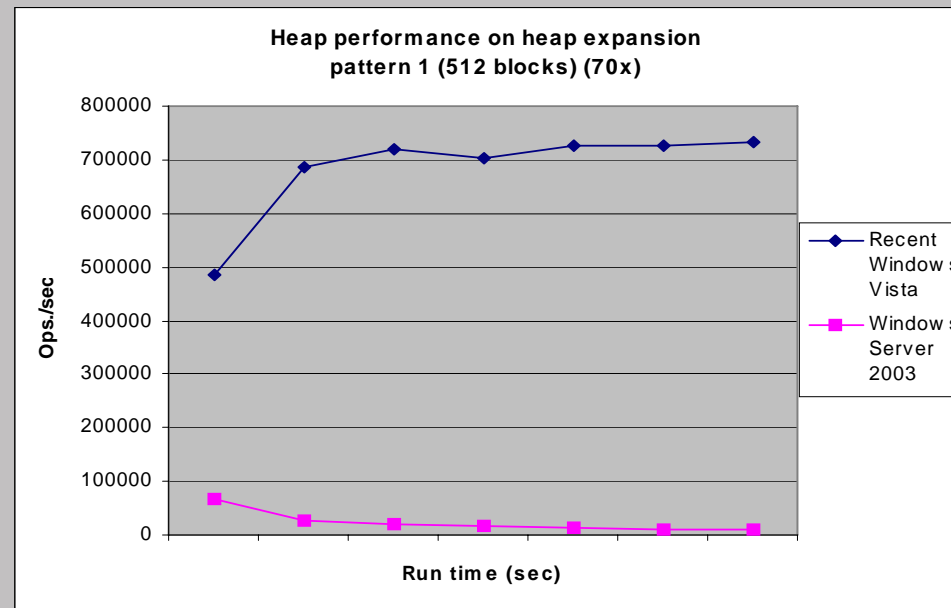


Fragmentation Test (512 blocks / 80 bytes)





Fragmentation Test (512 blocks / 80 bytes)

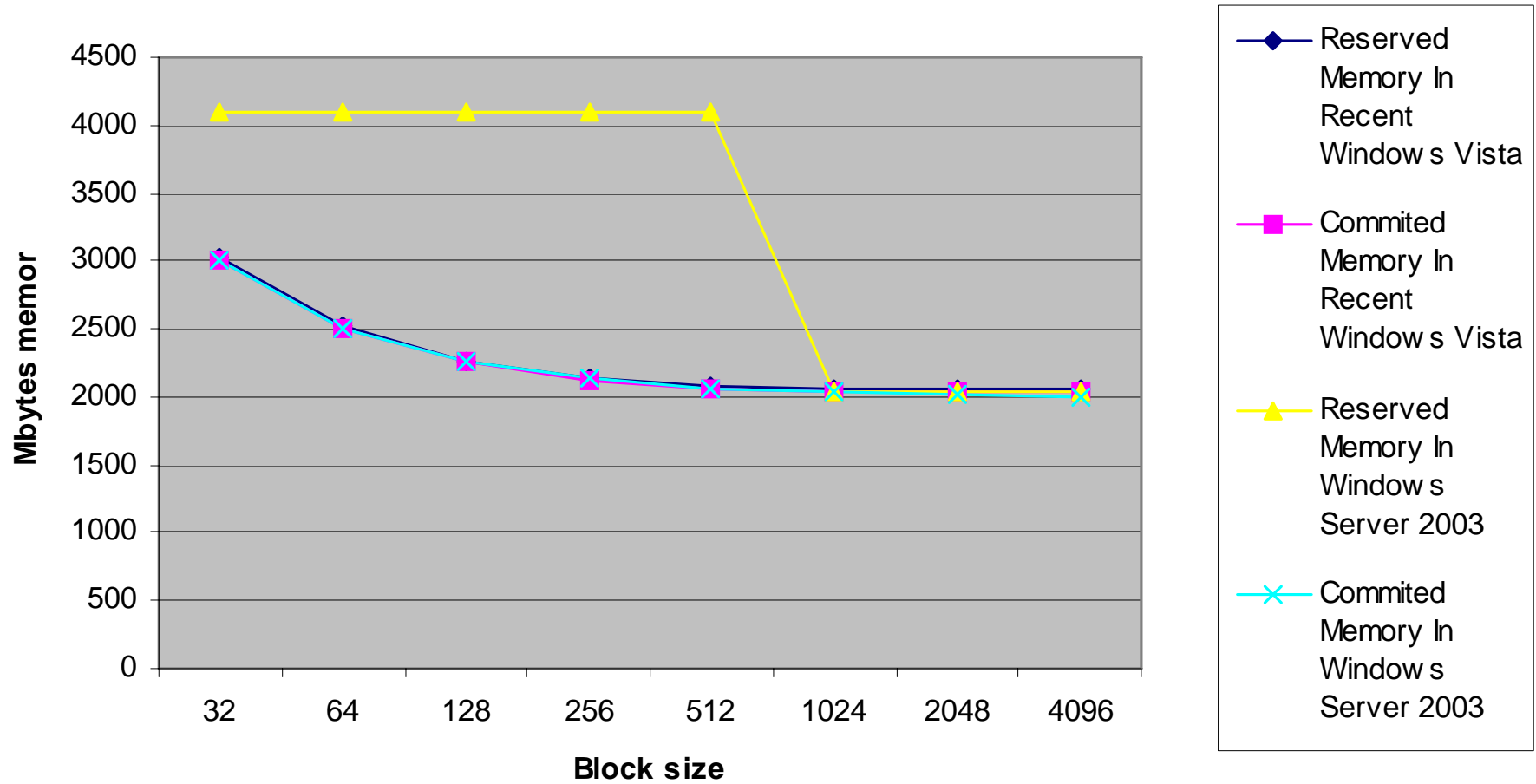




Fragmentation Scenario II

Pattern	Ops/sec (Recent Windows Vista)	Ops/sec (Windows Server 2003 SP 1)	Improveme nt x
256	2576004	388	6639
512	927709	151	6144
1024	403774	51	7917
2048	194180	25	7767
4096	82534	12	6878

Memory footprint on 2 GBytes heap expansion





Summary

- Attacks get more sophisticated ...
- But so does the heap management – and not only for security
- We laid the foundation for increased agility in heap improvements with reduced compatibility risks
- Improved scenarios for SMP and large memory usage
- Designed to enhance the code quality for applications
- We are not yet done ... we are looking forward for further enhancements as needed
- Come see me with your ideas!



Resources

- Feedback on Heap:
heapext@microsoft.com
- Debugging tools:
<http://www.microsoft.com/whdc/devtools/debugging/debugstart.mspx>
- Application Verifier:
<http://www.microsoft.com/downloads/details.aspx?FamilyID=bd02c19c-1250-433c-8c1b-2619bd93b3a2&DisplayLang=en>

X'COLL 2006



secure@microsoft.com

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.